



Universidad  
de Alcalá

COMISIÓN DE ESTUDIOS OFICIALES  
DE POSGRADO Y DOCTORADO

**ACTA DE EVALUACIÓN DE LA TESIS DOCTORAL**  
(FOR EVALUATION OF THE ACT DOCTORAL THESIS)

Año académico (academic year): 2016/17

DOCTORANDO (candidate PHD): **BRONTE PALACIOS, SEBASTIÁN**

PROGRAMA DE DOCTORADO (Academic Committee of the Programme): **D441-ELECTRÓNICA: SISTEMAS ELECTRÓNICOS AVANZADOS. SISTEMAS INTELIGENTES**

DEPARTAMENTO DE (Department): **ELECTRÓNICA**

TITULACIÓN DE DOCTOR EN (Phd title): **DOCTOR/A POR LA UNIVERSIDAD DE ALCALÁ**

En el día de hoy 11/07/17, reunido el tribunal de evaluación, constituido por los miembros que suscriben el presente Acta, el aspirante defendió su Tesis Doctoral **con Mención Internacional** (In today assessment met the court, consisting of the members who signed this Act, the candidate defended his doctoral thesis with mention as International Doctorate), elaborada bajo la dirección de (prepared under the direction of) LUIS MIGUEL BERGASA PASCUAL // DANIEL PIZARRO PÉREZ.

Sobre el siguiente tema (Title of the doctoral thesis): **REAL TIME SEQUENTIAL NON RIGID STRUCTURE FROM MOTION USING A SINGLE CAMERA**

Finalizada la defensa y discusión de la tesis, el tribunal acordó otorgar la CALIFICACIÓN GLOBAL<sup>1</sup> de (**no apto, aprobado, notable y sobresaliente**) (After the defense and defense of the thesis, the court agreed to grant the GLOBAL RATING (fail, pass, good and excellent): **SOBRESALIENTE**

Alcalá de Henares, a 11 de Julio de 2017

Fdo. (Signed): [Signature]

Fdo. (Signed): ARTURO DE LA ESCOBEDA

Fdo. (Signed): PABLO MESEJO

FIRMA DEL ALUMNO (candidate's signature),

[Signature]

Con fecha 24 de Julio de 2017 la Comisión Delegada de la Comisión de Estudios Oficiales de Posgrado, a la vista de los votos emitidos de manera anónima por el tribunal que ha juzgado la tesis, resuelve:

- ☒ Conceder la Mención de "Cum Laude"  
☐ No conceder la Mención de "Cum Laude"

La Secretaria de la Comisión Delegada

Fdo. (Signed): Sebastián Bronte Palacios

[Signature]

<sup>1</sup> La calificación podrá ser "no apto" "aprobado" "notable" y "sobresaliente". El tribunal podrá otorgar la mención de "cum laude" si la calificación global es de sobresaliente y se emite en tal sentido el voto secreto positivo por unanimidad. (The grade may be "fail" "pass" "good" or "excellent". The panel may confer the distinction of "cum laude" if the overall grade is "Excellent" and has been awarded unanimously as such after secret voting.)

INCIDENCIAS / OBSERVACIONES:  
(Incidents

/

Comments)

El presente informe tiene como objetivo  
informar a la Junta de Gobierno del  
Colegio de la existencia de un  
problema de mantenimiento en el  
edificio principal, concretamente en  
la zona de la cocina y el comedor.  
El problema consiste en la  
falta de agua caliente en  
estas zonas, lo que dificulta  
el funcionamiento normal del  
edificio y puede afectar a la  
salud de los alumnos y profesores.  
Se ha intentado solucionar el  
problema mediante la  
llamada al servicio de  
mantenimiento, pero no se ha  
podido solucionar hasta el  
momento. Se solicita a la Junta de  
Gobierno que tome las medidas  
necesarias para solucionar el  
problema lo antes posible.

En aplicación del art. 14.7 del RD. 99/2011 y el art. 14 del Reglamento de Elaboración, Autorización y Defensa de la Tesis Doctoral, la Comisión Delegada de la Comisión de Estudios Oficiales de Posgrado y Doctorado, en sesión pública de fecha 24 de julio, procedió al escrutinio de los votos emitidos por los miembros del tribunal de la tesis defendida por *BRONTE PALACIOS, SEBASTIÁN*, el día 11 de julio de 2017, titulada *REAL TIME SEQUENTIAL NON RIGID STRUCTURE FROM MOTION USING A SINGLE CAMERA*, para determinar, si a la misma, se le concede la mención "cum laude", arrojando como resultado el voto favorable de todos los miembros del tribunal.

Por lo tanto, la Comisión de Estudios Oficiales de Posgrado resuelve otorgar a dicha tesis la

***MENCIÓN "CUM LAUDE"***

Alcalá de Henares, 27 julio de 2017  
EL PRESIDENTE DE LA COMISIÓN DE ESTUDIOS  
OFICIALES DE POSGRADO Y DOCTORADO



Firmado digitalmente por VELASCO  
PEREZ JUAN RAMON - DNI  
03087239H  
Fecha: 2017.07.28 11:07:43 -06'00'

Juan Ramón Velasco Pérez

Copia por e-mail a:

Doctorando: BRONTE PALACIOS, SEBASTIÁN

Secretario del Tribunal: MARTA MARRÓN ROMERA.

Directores de Tesis: LUIS MIGUEL BERGASA PASCUAL // DANIEL PIZARRO PÉREZ



Universidad  
de Alcalá

ESCUELA DE DOCTORADO  
Servicio de Estudios Oficiales de  
Posgrado

DILIGENCIA DE DEPÓSITO DE TESIS.

Comprobado que el expediente académico de D./D<sup>a</sup> \_\_\_\_\_  
reúne los requisitos exigidos para la presentación de la Tesis, de acuerdo a la normativa vigente, y habiendo  
presentado la misma en formato: ☐ soporte electrónico ☐ impreso en papel, para el depósito de la  
misma, en el Servicio de Estudios Oficiales de Posgrado, con el nº de páginas: \_\_\_\_\_ se procede, con  
fecha de hoy a registrar el depósito de la tesis.

Alcalá de Henares a \_\_\_\_\_ de \_\_\_\_\_ de 20\_\_\_\_



Fdo. El Funcionario





PhD. Program in Electronics: Advanced Electronic  
Systems. Intelligent Systems

# **Real Time Sequential Non Rigid Structure from Motion Using a Single Camera**

PhD. Thesis Presented by  
**Sebastián Bronte Palacios**

2017





PhD. Program in Electronics: Advanced Electronic  
Systems. Intelligent Systems

# **Real Time Sequential Non Rigid Structure from Motion Using a Single Camera**

PhD. Thesis Presented by  
**Sebastián Bronte Palacios**

Advisor  
**Dr. Luis Miguel Bergasa Pascual**

Co-Advisor  
**Dr. Daniel Pizarro Pérez**

Alcalá de Henares, 2017





Universidad  
de Alcalá

DEPARTAMENTO DE ELECTRÓNICA  
Edificio Politécnico  
Campus Universitario s/n  
28805 Alcalá de Henares (Madrid)  
Teléfono: 91 885 65 40  
Fax: 91 885 65 91  
eldep@depeca.uah.es

Dr. Luis Miguel Bergasa Pascual, Catedrático de la Universidad de Alcalá

Dr. Daniel Pizarro Pérez, Profesor Titular de Universidad Interino

INFORMA:

Que la Tesis Doctoral titulada "Real Time Sequential Non Rigid Structure from motion using a single camera", presentada por D. Sebastián Bronte Palacios, y realizada bajo nuestra dirección, dentro del campo del modelado de objetos deformables basado en imágenes obtenidas desde una cámara, reúne los méritos de calidad y originalidad para optar al Grado de Doctor.

Alcalá de Henares, a 27 de abril de 2017.

Fdo.: Luis Miguel Bergasa Pascual.



Fdo.: Daniel Pizarro Pérez.







Dra. Sira Palazuelos Cagigas, Directora del Departamento de Electrónica de la Universidad de Alcalá,

INFORMA:

Que la Tesis Doctoral titulada "Real Time Sequential Non Rigid Structure from motion using a single camera", presentada por D. Sebastián Bronte Palacios, y dirigida por el Dr. Luis Miguel Bergasa Pascual y el Dr. Daniel Pizarro Pérez, cumple con todos los requisitos científicos y metodológicos para ser defendida ante un Tribunal.

Alcalá de Henares, a 27 de abril de 2017.



Fdo.: Sira Palazuelos Cagigas.



**A mis padres, familia y novia...**

*“Don’t ever let somebody tell you ... you can’t do something. Not even me. Alright? You got a dream ... You gotta protect it. People can’t do something themselves, they wanna tell you you can’t do it. If you want something, go get it. Period.”*

Christopher Gardner. The pursuit of happyness. 2006

*“Nunca dejes que nadie te diga que no puedes hacer algo. Ni siquiera yo, ¿vale? Si tienes un sueño ... tienes que protegerlo. Las personas que no son capaces de hacer algo te dirán que tú tampoco puedes. Si quieres algo ve por ello. Punto.”*

Christopher Gardner. En busca de la felicidad. 2006



# Agradecimientos /

## Acknowledgements

Quiero agradecer a mi tutor, Luis Miguel Bergasa por la paciencia que ha tenido conmigo a lo largo de estos años. Durante todo este largo tiempo de reuniones, la constancia de mantenerlas cada 2 semanas ha sido lo que ha mantenido la tensión para poder terminar. Además he de agradecer las útiles revisiones de los artículos publicados, tanto del IROS como los del IV, ITSC y otros proyectos con los compañeros, y por supuesto la intensa, durísima, pero siempre necesaria, útil y constructiva revisión de la tesis.

Special mention to Lourdes Agapito for allowing me to do an internship with her, as that was the actual starting point of this PhD thesis. Thanks to her, I had also the opportunity to meet her students, not only from the the academic perspective, but also from a more personal point of view, as, since 2011 we are still in touch: Marco and his wife Maria, Franz (thanks so much to wellcome me “that day”), Tassos and Marianthi, Ravi, Nikos, Chris, Matteo and, of course A. Davison.

He de agradecer también a Daniel Pizarro su inestimable ayuda dados sus profundos conocimientos en el tema en los últimos momentos de esta tesis, incluso estando a punto de ser padre. Aún en esos momentos, y queriendo echar una mano para que esta tesis llegara a buen puerto, es algo que tengo muy en cuenta.

Aunque pueda parecer una broma, quisiera agradecer a Renfe la labor de llevarme y traerme al trabajo mientras estaba desarrollando algunos de los puntos o estaba escribiendo algún paper. También he de reconocer por esta labor a los que han compartido coche conmigo para ir al trabajo, lo que me ha permitido picar código mientras estaba en los trayectos.

De manera genérica podría sonar banal, pero gracias a la música he podido pasar buenos y menos buenos momentos, pero sobre todo, he podido concentrarme un poco mejor y entrar, como su nombre de género indica, en un estado de “Trance”. Es por ello que merecen unas líneas en esta tesis artistas como Armin van Buuren, Ferry Corsten, Borja Iglesias, Hazem Beltagui, Above & Beyond, Robert Miles, Arksun, y un largo etc. por creaciones que me ayudan a trabajar con más ganas.

Por supuesto a los amigos que siempre han estado ahí y que estos últimos años no he

podido atender por la “maldita tesis”: Fran y Cris, Adri y Elena, Santi y Elena, Dani y Belen, Edu, Murcia, Manzanares y Maricruz, Raúl, Sara, Alberto, Luis, Oscar, Antero, Raúl y Patri, Dave y Toñi, Jose Luis, Juanpa, Miguel, Amaia, Madrigal, Esteban, Semper, Aida y Harry, Irene, Antonio y Silvia, David, Enrique, Sergio, Susana, Andrea, Diego, Marta (y podría seguir y seguir ...). Por supuesto, los amigos que da la casualidad que son también compañeros con los que he tenido el gusto de compartir estudios de doctorado: Pablo, Javi Yebes, Rober, Los Edus, Oscar, Iván, Álvaro, Jesús, Llama, Raúl, Miguel, Fer, Noe, Nacho, y otros tantos que han pasado por el laboratorio. Mención especial para Lidia y Balky por intentar echarme una mano con este proyecto.

Además, están aquellos que he ido conociendo en los trabajos por los que he ido pasando a lo largo de estos duros años de compaginar que han estado ahí y han sabido entender la magnitud de lo que llevaba encima: En GMV: Jose Luis, Gabi, Hector, Jose Ignacio, Alex, Josema, Luis, Raquel, Antonio, Fran, Johnathan, ... En Sepsa a todos porque éramos una piña, pero en especial a Antonio y finalmente en Altran/Ericsson a Jose Javi, Elías, Javi Muñoz, Jose Garvayo, Jorge, Dani Torres, Dani Tejedor, los Nachos, Juan, Jose Ángel, Álvaro, David, Jose Luis y un largo etc. Carlos Soler merece mención a parte por dejarme realizar teletrabajo durante los últimos meses y así ganar tiempo. Si no me he vuelto más loco o no he desistido por agotamiento ha sido en parte gracias a él.

No podía terminar esta sección con los responsables de que todo esto sea posible: mis padres. Nunca han dejado de animarme y empujarme aunque a veces quisiera abandonar por la presión, además de que en la reforma de la casa fuesen los que más han estado ahí. Son los pilares máximos en los que me he apoyado y que últimamente no he atendido como merecieran, aunque han sabido entenderlo. Eternamente agradecido quedo por ello. El resto de la familia también ha sufrido las consecuencias, puesto que apenas he podido verla en los últimos años. También recordar a los que, mientras he estado trabajando en esto no van a poder compartir la alegría de haber terminado: mi abuela y mi tía.

Debo reservar el último párrafo al último pilar: mi pareja, Elva. Aunque al principio fuese duro, con una reforma de por medio, al final me ha ayudado muchísimo. Desde que la conozco me ha hecho la vida mucho más fácil, bonita y ha sido capaz de amortiguar los momentos difíciles, en los que me ha ayudado en lo posible para que yo pudiese emplear el poco tiempo que tenía de manera más efectiva y ayudar a centrarme. Mil gracias por todo. Te quiero.

*La vida es aquello que te  
va sucediendo mientras te  
empeñas en hacer otros planes*

John Lennon.



# Resumen

Las aplicaciones que basan su funcionamiento en una correcta localización y reconstrucción dentro de un entorno real en 3D han experimentado un gran interés en los últimos años, tanto por la comunidad investigadora como por la industrial. Estas aplicaciones cubren desde la realidad aumentada, la robótica, la simulación, los videojuegos, etc. Dependiendo de la aplicación y del nivel de detalle requerido en la reconstrucción, se emplean diversos dispositivos como: cámaras estéreo, sensores Red Green Blue and Depth (RGBD) con Luz estructurada, cámaras Time of Flight (ToF), láseres 2D / 3D, etc. En los casos de aplicaciones más sencillas se pueden usar dispositivos de uso común, como los smartphones, en los que aplicando técnicas de visión artificial, se pueden obtener modelos 3D del entorno de trabajo con suficiente calidad para mostrar información aumentada.

En robótica, la localización y generación simultáneas de un mapa del entorno en 3D a partir de una cámara es una tarea fundamental para conseguir la navegación autónoma de robots. Para ello se han utilizado técnicas conocidas en el estado del arte como Simultaneous Localization And Mapping (SLAM) o Structure from Motion (SfM). La condición para la aplicación de estas técnicas es que el objeto de interés no cambie su forma a lo largo del tiempo, esto es, sea rígido. La reconstrucción es unívoca a falta de un factor de escala, que en una captura monocular no es posible obtener sin una referencia fija.

Si la condición de rigidez en la escena no se cumple, es porque la forma del objeto cambia a lo largo del tiempo, luego es deformable. Por tanto, el problema sería equivalente a realizar una reconstrucción por fotograma, lo cual no se puede hacer de manera directa, sino que hay que recurrir a otros métodos. Además, el problema, así planteado, es muy ambiguo, puesto que diferentes formas, combinadas con diferentes poses de cámara pueden dar lugar a proyecciones similares. Es por esto que el campo de la reconstrucción de objetos deformables es todavía un área en desarrollo. Los métodos de SfM se han ido adaptando a la reconstrucción de objetos deformables aplicando modelos físicos, restricciones temporales, espaciales, geométricas o de otros tipos para reducir la ambigüedad en las soluciones, naciendo así las técnicas conocidas como Non-Rigid Structure from Motion (NRSfM).

En esta tesis se propone partir de una técnica de reconstrucción rígida bien conocida en el estado del arte como es PTAM (Parallel Tracking and Mapping) y adaptarla para que sea capaz de incluir técnicas de NRSfM, basadas en modelo de bases lineales para estimar

las deformaciones del objeto modelado dinámicamente y aplicar restricciones temporales y espaciales para mejorar las reconstrucciones, además de ir adaptándose a cambios de deformación que se presenten en la secuencia. Los problemas de asociación de datos sobre imágenes reales son también abordados.

Para ello, ha habido que realizar cambios en PTAM de manera que cada uno de sus hilos de ejecución (seguimiento y generación de mapa) pasasen a procesar los datos no rígidos de manera natural.

El hilo encargado del seguimiento ya realizaba de manera nativa seguimiento basado en un mapa de puntos 3D, proporcionado a priori. La modificación más importante propuesta para este hilo es la integración de un modelo de deformación lineal para que se realice el cálculo de la deformación del objeto en tiempo real, asumiendo fijas las formas básicas de deformación. El cálculo de la pose de la cámara está basado en el sistema de estimación rígido, por lo que la estimación de pose y coeficientes de deformación se hace de manera alternada usando el algoritmo E-M (Expectation-Maximization). También, se imponen restricciones temporales y de forma para minimizar las ambigüedades inherentes en las soluciones y mejorar la calidad de la estimación 3D.

Respecto al hilo que gestiona el mapa, se actualiza en función del tiempo para que sea capaz de mejorar las bases de deformación cuando éstas no son capaces de explicar las formas que se ven en las imágenes actuales. Para ello, se sustituye la técnica de optimización del modelo rígido, Sparse Bundle Adjustment (SBA), por un método de procesamiento exhaustivo no rígido NRSfM para mejorar las bases acorde a las imágenes con gran error de reconstrucción que llegan desde el hilo de seguimiento. Con esto, el modelo se consigue adaptar a nuevas deformaciones de manera secuencial, permitiendo al sistema evolucionar y ser estable a largo plazo.

A diferencia de una gran parte de los métodos de la literatura, el sistema propuesto aborda el problema de la proyección perspectiva de forma nativa, minimizando los problemas de ambigüedad y de distancia al objeto existente en la proyección ortográfica. El sistema propuesto maneja centenares de puntos y está preparado para cumplir con las restricciones de tiempo real necesarias para su aplicación en sistemas con recursos hardware limitados. Además, presenta un buen equilibrio entre error de reconstrucción y tiempo de procesamiento respecto a otras propuestas del estado del arte.

**Palabras clave:** **model-based non-rigid reconstruction, NRSfM, SfM, AR, PTAM descriptores, reconstrucción 3D, objetos deformables NRSfM, Reconstrucción 3D, seguimiento basado en modelo, visión artificial, PTAM.**

# Abstract

There are applications based in a correct localization and reconstruction of a scene in a real 3D environment, which has experienced a great interest in the latest years by researchers and industrial community. These applications cover from augmented reality, robotics, simulation, video-games, etc. Depending on the application and the required reconstruction detail level, different devices can be used such as: stereo cameras, Red Green Blue and Depth (RGBD) sensors using Structured Light, Time of Flight (ToF) cameras, 2D / 3D lasers, etc. Simpler applications can use less complex hardware, i.e. commonly use devices, like smartphones, and applying computer vision techniques, 3D models of the workspace can be obtained with quality enough to render augmented information.

In robotics, localization and simultaneous 3D map generation using a camera is a fundamental task for autonomous navigation. To that end, Simultaneous Localization And Mapping (SLAM) or Structure from Motion (SfM) techniques have been used. The condition for applying these techniques is the target object must not change its shape along the time, so it must be rigid. In this case, the reconstruction is unique up to scale, given that for a monocular capture is not possible to recover it unless there is a fixed reference.

In case the rigidity condition does not apply on the scene, the object changes its shape along the time, so it is deformable. Therefore the problem would be equivalent to perform a reconstruction per frame, which is an ill posed problem and so ambiguous, as different shapes combined with certain camera poses could lead to similar projections. This is why deformable object reconstruction is an active research field nowadays. To perform the reconstructions, SfM methods have been adapting to the non-rigid reconstruction of deformable objects by incorporating physical models, temporal, spacial and geometrical priors or other kinds of restrictions to reduce the solutions and better conform the reconstruction, giving as a result the Non-Rigid Structure from Motion (NRSfM) techniques.

In this Thesis, we propose to depart from a well known state-of-the-art technique PTAM (Parallel Tracking and Mapping) and adapt it to include NRSfM techniques, based on linear bases model to estimate the object deformations dynamically and apply temporal and spacial restrictions to improve the reconstruction. Additionally it is modified to adapt to changes on the deformation types of the sequence.

To that end, there has been changes to be applied to each of PTAM execution threads

to process the incoming non-rigid data of the scene in a natural way. Data association problems are faced as well.

The tracking thread was already doing tracking from template in a native way, based on 3D map points, previously provided. The main modification proposal of this thread is the integration of a linear shape bases model to perform the computation of the shape deformations in real time assuming the deformation bases fixed. The pose computation is based on the previous rigid estimation system, so the whole state estimation is done alternating pose and deformation coefficient steps by using an Expectation-Maximization (EM) algorithm. Temporal and shape smoothness priors are also imposed to minimize the ambiguities inherent to the solutions and to improve the 3D estimations quality.

Regarding the mapping thread, it is modified so that it can handle deformation bases improvements when the current set of bases are not able to explain the currently seen deformations on the image. To that end, the rigid optimization technique of Sparse Bundle Adjustment (SBA) is substituted by an exhaustive non-rigid NRSfM batch algorithm to improve the bases according to the images having a great *reprojection* error that are sampled from the tracking thread. With this setup, we are able to adapt to new deformations in a sequential way, allowing the system to evolve and being stable in the long term.

Unlike some literature methods, the proposed system faces the perspective problem in a native way, minimizing the problems of the ambiguity on the distance to the object existing with the orthographic projection approaches. The proposed system also handles hundreds of points and is ready to comply with real-time restrictions for its application on limited hardware resources systems. Additionally, it presents a good trade-off between reconstruction error and processing time regarding other proposals of the state-of-the-art.

**Keywords:** model-based non-rigid reconstruction, NRSfM, SfM, AR, PTAM, descriptors, 3D reconstruction, deformable objects NRSfM, 3D reconstruction, Model-based tracking, Computer vision, PTAM.

# Contents

<b>Resumen</b>	<b>xiii</b>
<b>Abstract</b>	<b>xv</b>
<b>Contents</b>	<b>xvii</b>
<b>List of Figures</b>	<b>xxi</b>
<b>List of Tables</b>	<b>xxv</b>
<b>List of Acronyms</b>	<b>xxviii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem description . . . . .	1
1.2 Structure from Motion (SfM) . . . . .	2
1.3 Non Rigid Structure from Motion (NRSfM) . . . . .	2
1.4 Shape-from-Template (SfT) . . . . .	4
1.5 Motivation . . . . .	4
1.6 Applications . . . . .	5
1.6.1 Augmented reality, apps and games . . . . .	5
1.6.2 Film industry / Cinema . . . . .	7
1.6.3 Marketing . . . . .	7
1.6.4 Surgery training / simulation. Medical imaging . . . . .	8
1.6.5 Material research / Modelling . . . . .	9
1.6.6 Arts . . . . .	9

<b>2</b>	<b>Related work</b>	<b>11</b>
2.1	Structure from Motion (SfM)	12
2.1.1	Sparse reconstruction	12
2.1.1.1	Batch approaches	12
2.1.1.2	Sequential approaches	13
2.1.2	Parallel Tracking and Mapping (PTAM)	14
2.1.2.1	The tracking thread	15
2.1.2.2	The mapping thread	17
2.1.3	Dense reconstruction	21
2.2	Non-Rigid Reconstructions	23
2.2.1	Study of the ambiguities and deformation priors	23
2.2.1.1	Physics-based priors	25
2.2.1.2	Statistics-based priors	26
2.2.2	Model-Based reconstruction techniques	27
2.2.2.1	Statistics-based SfT	27
2.2.2.2	Physics-based Shape from Template (SfT)	28
2.2.3	Non-Rigid Structure-from-Motion (NRSfM)	30
2.2.3.1	Batch NRSfM	30
2.2.3.2	Sequential NRSfM	31
2.2.3.3	Camera models in NRSfM	31
2.2.3.4	Bregler et al.'s batch approach	33
2.2.3.5	Paladini et al.'s Sequential approach	35
2.3	Thesis objectives	38
2.3.1	Non rigid tracking	38
2.3.2	Incremental Modeling	38
2.3.3	Parallel tracking and modelling on collaborative mode.	38
2.3.4	Perspective projection	39
2.3.5	Dealing with real data association	39
<b>3</b>	<b>Real-Time Model-Based Non-Rigid Tracking</b>	<b>41</b>
3.1	Algorithm Description	42
3.1.1	Measurement model / data association	44
3.1.2	Feature matching	45



3.1.2.1	PTAM based matching approach . . . . .	45
3.1.2.2	Descriptor based matching approach . . . . .	46
3.1.3	Motion model . . . . .	46
3.1.4	E-M optimization . . . . .	47
3.1.4.1	E-Step. Deformation estimation . . . . .	47
3.1.4.2	M-Step, Pose estimation . . . . .	49
3.1.5	Priors . . . . .	52
3.1.6	Tracking recovery . . . . .	53
3.2	Results . . . . .	54
3.2.1	Performance metrics . . . . .	54
3.2.2	Sequences . . . . .	57
3.2.2.1	CMUface Sequence . . . . .	57
3.2.2.1.1	Performance evaluation based on perfect matching	57
3.2.2.1.2	Performance evaluation based on visibility degradation . . . . .	61
3.2.2.1.3	Performance evaluation based on noise and outliers	63
3.2.2.1.4	Performance evaluation based on the number of bases . . . . .	64
3.2.2.1.5	Comparison with other methods of the state-of-the-art . . . . .	65
3.2.2.2	Flag Sequence . . . . .	66
3.2.2.2.1	Performance evaluation based on perfect matching	67
3.2.2.2.2	Performance evaluation based on visibility degradation . . . . .	70
3.2.2.2.3	Performance evaluation based on noise and outliers	71
3.2.2.2.4	Performance evaluation based on the number of bases . . . . .	72
3.2.2.2.5	Comparison with other methods of the state-of-the-art . . . . .	73
3.2.2.3	Rendered Flag sequence . . . . .	75
3.2.2.3.1	Visual descriptor evaluation . . . . .	77
3.2.2.3.2	Performance evaluation based on time and shape priors . . . . .	81

3.2.2.3.3	Performance evaluation based on the number of bases . . . . .	93
3.2.2.3.4	Performance evaluation based on tracking loss recovery . . . . .	96
3.3	Conclusions . . . . .	102
<b>4</b>	<b>Sequential keyframe-Based Non-Rigid Mapping</b>	<b>103</b>
4.1	General architecture . . . . .	104
4.2	Sequential modeling validation proposal . . . . .	105
4.3	Results . . . . .	107
4.3.1	Experiment setup . . . . .	107
4.3.2	Test Sequence: Rendered flag sequence . . . . .	109
4.4	Conclusions . . . . .	121
<b>5</b>	<b>Conclusions and Future Works</b>	<b>123</b>
5.1	Conclusions and contributions . . . . .	123
5.2	Future Works . . . . .	125
	<b>Bibliography</b>	<b>129</b>
<b>A</b>	<b>Translation-size ambiguity study</b>	<b>141</b>
<b>B</b>	<b>Prior derivation</b>	<b>145</b>
B.1	Temporal smoothness . . . . .	145
B.2	Spatial smoothness . . . . .	146
B.3	Isometry . . . . .	147
B.4	Aggregation of priors on the tracking solvers . . . . .	147
<b>C</b>	<b>Closed form coefficient deduction</b>	<b>149</b>
C.1	Rigid shape is a function of the bases . . . . .	149
C.2	Taking into account rigid shape on the estimation . . . . .	151

# List of Figures

1.1	Drakerz and Vuforia . . . . .	5
1.2	Kinect and Skeleton Tracking example . . . . .	6
1.3	Augmented Reality (AR) / Virtual Reality (VR) applications . . . . .	6
1.4	Apps using SfM / NRSfM . . . . .	7
1.5	AR on PSVita games . . . . .	7
1.6	Motion Capture system examples . . . . .	7
1.7	AR marketing examples . . . . .	8
1.8	AR Surgery examples . . . . .	8
2.1	PTAM mapping workflow diagram . . . . .	19
2.2	PTAM incremental mapping example . . . . .	20
2.3	Extended Kalman Filter (EKF)-SLAM vs PTAM comparison . . . . .	21
2.4	Dense Tracking And Mapping (DTAM) vs PTAM tracking comparison for fast movements . . . . .	22
2.5	DTAM inverse depth map vs PTAM sparse mapping . . . . .	22
2.6	Kinect Fusion examples . . . . .	23
2.7	Ambiguity illustration example . . . . .	25
2.8	Geometric modelling scheme for isometric SfT . . . . .	28
2.9	Linear basis shapes approximation . . . . .	30
2.10	Finite Element Method (FEM) modelling principles . . . . .	32
2.11	Orthographic vs perspective projection . . . . .	32
2.12	Sequential NRSfM results on actress sequence . . . . .	37
3.1	Tracking thread block diagram . . . . .	43
3.2	Tracking data association . . . . .	44
3.3	CMUface sequence snapshots for certain frames . . . . .	58

3.4	CMUface sequence reconstruction examples without tracking degradation .	59
3.5	Trajectory estimation for CMUface sequence without tracking degradation	60
3.6	CMUface sequence reprojection and 3D reconstruction error over time for no degradation tracking . . . . .	60
3.7	CMUface sequence screenshots for visibility experiments . . . . .	61
3.8	CMUface sequence error results for visibility degradation on tracking . . .	62
3.9	CMUface sequence screenshots for noise and outlier experiments . . . . .	63
3.10	CMUface sequence error results on noise and outliers degradation on tracking	64
3.11	CMUface sequence Error over time varying the number of bases . . . . .	65
3.12	Flag sequence snapshots for certain frames . . . . .	67
3.13	Flag sequence reconstruction examples without tracking degradation . . . .	68
3.14	Trajectory estimation for Flag sequence without tracking degradation . . .	69
3.15	Flag sequence representation of the errors over time. At the left the repro- jection error and at the right the 3D reconstruction error . . . . .	69
3.16	Flag sequence screenshots for visibility experiments . . . . .	70
3.17	Flag sequence results for visibility degradation on tracking . . . . .	71
3.18	Flag sequence screenshots for noise and outlier experiments . . . . .	72
3.19	Flag sequence results for noise and outliers degradation on tracking . . . .	73
3.20	Flag sequence error over time varying the number of bases . . . . .	74
3.21	Rendered flag sequence screenshots . . . . .	76
3.22	Rendered flag sequence reconstruction results for different descriptors . . .	78
3.23	Rendered flag sequence rank based on the error results of the different descriptor methods . . . . .	80
3.24	Trajectory estimation for Rendered flag sequence. SIFT descriptor example	80
3.25	Rendered flag sequence screenshots when priors are applied for <i>IROS</i> -based tracking . . . . .	82
3.26	Rendered flag sequence screenshots when priors are applied for Scale In- variant Feature Transform (SIFT) descriptor . . . . .	83
3.27	Rendered flag sequence reconstruction for <i>IROS</i> descriptors . . . . .	84
3.28	Rendered flag sequence reconstruction results for SIFT descriptors . . . . .	85
3.29	Rendered flag sequence reprojection error results over time for time smooth- ness prior . . . . .	87
3.30	Rendered flag sequence reconstruction error results over time for time smoothness prior . . . . .	87

3.31	Rendered flag sequence rotation error results over time for time smoothness prior . . . . .	88
3.32	Rendered flag sequence translation error results over time for time smoothness prior . . . . .	88
3.33	Rendered flag sequence reprojection error results over time for shape smoothness prior . . . . .	89
3.34	Rendered flag sequence reconstruction error results over time for shape smoothness prior . . . . .	89
3.35	Rendered flag sequence rotation error results over time for shape smoothness prior . . . . .	90
3.36	Rendered flag sequence translation error results over time for shape smoothness prior . . . . .	90
3.37	Trajectory representation for Rendered flag sequence for different time and shape priors . . . . .	92
3.38	Rendered flag sequence 2D reprojection error over time varying the number of bases . . . . .	94
3.39	Rendered flag sequence 3D reconstruction error over time varying the number of bases . . . . .	95
3.40	Rendered flag sequence rotation error over time varying the number of bases . . . . .	95
3.41	Rendered flag sequence translation error over time varying the number of bases . . . . .	96
3.42	Rendered flag sequence analysis over time of tracking quality for different descriptors . . . . .	99
3.43	Rendered flag sequence temporal 2D error analysis for tracking loss . . . . .	100
3.44	Rendered flag sequence temporal 3D reconstruction error analysis for tracking loss . . . . .	100
3.45	Rendered flag sequence rotation error analysis for tracking loss and different descriptors . . . . .	101
3.46	Rendered flag sequence translation error analysis for tracking loss and different descriptors . . . . .	101
4.1	Architecture of the main mapping thread modules . . . . .	104
4.2	Screenshots for flag sequence processing (first fourth) . . . . .	111
4.3	Screenshots for flag sequence processing (second fourth) . . . . .	112
4.4	Screenshots for flag sequence processing (third fourth) . . . . .	113
4.5	Screenshots for flag sequence processing (four fourth) . . . . .	114

4.6	3D reconstruction representations for flag sequence (first fourth) . . . . .	115
4.7	3D reconstruction representations for flag sequence (second fourth) . . . . .	116
4.8	3D reconstruction representations for flag sequence (third fourth) . . . . .	117
4.9	3D reconstruction representations for flag sequence (four fourth) . . . . .	118
4.10	2D reprojection error over time for visibility and outlier rejection when regenerating the bases . . . . .	119
4.11	3D reconstruction error over time for visibility and outlier rejection when regenerating the bases . . . . .	119
4.12	rotation error over time for visibility and outlier rejection when regenerat- ing the bases . . . . .	120
4.13	Translation error over time for visibility and outlier rejection when regen- erating the bases . . . . .	120
B.1	Shape smoothness graphical representation . . . . .	146



# List of Tables

2.1	Rigid SfM approaches summary . . . . .	24
2.2	Model-based / SfT approaches summary . . . . .	29
2.3	NRSfM approaches summary . . . . .	33
3.1	CMUface sequence error comparison with the number of bases . . . . .	65
3.2	CMUface results summary against other methods . . . . .	66
3.3	Flag sequence error comparison with the number of bases . . . . .	73
3.4	Flag sequence results summary against other methods . . . . .	75
3.5	Rendered flag sequence descriptor results comparative . . . . .	79
3.6	Results summary for different prior types and the influence of the descriptor type used . . . . .	91
3.7	Rendered flag sequence performance vs the number of bases . . . . .	93



# List of Acronyms

3DMM    3D Morphable Models.

AAM    Active Appearance Models.

AR    Augmented Reality.

ARAP    As Rigid As Possible.

BA    Bundle Adjustment.

BRISK    Binary Robust Invariant Scalable Keypoints.

CAD    Computer Aided Design.

DoF    Degrees of Freedom.

DTAM    Dense Tracking And Mapping.

EKF    Extended Kalman Filter.

EM    Expectation-Maximization.

FAST    Features from Accelerated Segment Test.

FEM    Finite Element Method.

FFD    Free Form Deformations.

HMI    Human-Machine Interface.

ICP    Iterative Closest Point.

IR    Infra Red.

LIDAR    Light Detection and Ranging.

LLS    Linear Least Squares.

LM	Levenberg-Marquard.
MLE	Maximum Likelihood Estimation.
MP	Metric Projection.
MSM	Mass Spring Model.
MVS	Multi View Stereo.
NRSfM	Non-Rigid Structure from Motion.
ORB	Oriented FAST and Rotated BRIEF.
PCA	Principal Component Analysis.
PDE	Partial Differential Equation.
PTAM	Parallel Tracking And Mapping.
RANSAC	RANdom SAmple Consensus.
RGBD	Red Green Blue and Depth.
SAM	Smoothing And Mapping.
SBA	Sparse Bundle Adjustment.
SfM	Structure from Motion.
SfT	Shape from Template.
SIFT	Scale Invariant Feature Transform.
SLAM	Simultaneous Localization And Mapping.
SOCP	Second Order Cone Programming.
SURF	Speeded Up Robust Features.
SVD	Singular Value Decomposition.
ToF	Time of Flight.
TV	Total Variation.
VR	Virtual Reality.
WLS	Weighted Least Squares.

# Chapter 1

## Introduction

### 1.1 Problem description

The problem of 3D reconstruction and camera localization from images is known as Structure from Motion (SfM). 3D awareness from visual cues is a natural task for a human being, yet it is a very challenging problem in computer vision. During the last decades, SfM has been widely studied. Thanks to modern computing capabilities (multicore CPU's, GPUs, etc), current SfM algorithms are considered mature and capable of dealing with big amounts of data at standard video frame rates.

The general assumption in SfM is the rigidity of the environment, where changes in the images are due to relative motion between the camera and the scene. Rigidity links camera motion with image motion, making SfM a well-posed problem. Rigid SfM fails in scenarios where the rigidity assumption is violated. For instance, it fails to reconstruct scenes with multiple objects moving independently or with deformable objects that change their shape with time, such as the human body, articulated bodies, wires, flags, sheets, flesh, fabric, etc.

Reconstruction of deformable objects from images is known as Non-Rigid Structure from Motion (NRSfM) and has been actively studied in the recent years. Current NRSfM methods lack the level of maturity of SfM and it is a field under constant development.

In both SfM and NRSfM, there are two main categories of methods: *Batch approaches* and *Sequential approaches*. In the former, all data (images) are available beforehand, and jointly processed to obtain 3D. This approach is typically highly demanding in terms of time and memory but achieves accurate reconstruction results. In the latter, the data is collected and processed online. This is usually a harder problem where less data is available for reconstruction than in batch approaches. This results in less accurate reconstructions. However, online methods open the possibility of real-time applications that need sequential reconstructions, such as Augmented Reality (AR).

## 1.2 Structure from Motion (SfM)

Structure from Motion can be defined as the problem of jointly inferring the 3D geometry of a scene and the camera motion using images as inputs. Rigidity of the scene is a prior condition for SfM. The geometry of multiple views of a rigid scene has been known for centuries and the basic results for SfM are well known in photogrammetry and computer vision [Hartley and Zisserman, 2004]. Modern SfM methods cope with large sequences from both uncalibrated and calibrated cameras.

Sequential approaches in SfM are very important in robotics and were mainly developed as solutions to the visual Simultaneous Localization And Mapping (SLAM) problem, where the robot's pose and a map of the 3D environment is sequentially obtained from a camera system mounted on a mobile robot.

At the beginning, cameras were not the most important sensor to generate and locate the robot inside a map. Given the improvements of SfM algorithms, vision based SLAM became predominant in robotics, usually fused with other sensing elements installed in the robot.

One of the first monocular visual SLAM systems was proposed in [Davison et al., 2007]. It was posed as a sequential Bayesian inference problem, using the Extended Kalman Filter (EKF) as the inference core and a sparse set of salient feature tracks as observations. This method was real-time in a low cost hardware and was suitable in small to medium size environments.

Years later another sparse SLAM method was proposed, based on sequential Bundle Adjustment and known as Parallel Tracking And Mapping (PTAM) [Klein and Murray, 2007]. This method showed very accurate and stable pose estimation and reconstructions, suitable for Augmented Reality (AR) applications. It remarkably improved over SLAM methods based on statistical filtering. PTAM heavily influenced modern SLAM methods and changed the processing paradigm from pure sequential to a parallel mapping and tracking algorithm.

Recently, dense SfM methods have been proposed, such as Dense Tracking And Mapping (DTAM) [Newcombe et al., 2011b], further extended to depth sensors, such as Kinect fusion [Newcombe et al., 2011a]. Using this approach, multiple object tracking on a scene was developed, as well as dense dynamic scene tracking.

## 1.3 Non Rigid Structure from Motion (NRSfM)

In NRSfM the objective is to recover the 3D shape of an object undergoing deformations from a sequence of images. Each image shows the combination of rigid motion and shape change in the object and thus rigid SfM is not applicable in this case. NRSfM is ill-posed unless priors on the possible deformations are imposed. According to the

deformation prior, existing NRSfM methods can be divided into two main groups: *i) Physics-based* and *ii) Statistical-based* methods. In *Physics-based* models, the deformation model is taken from the field of *Continuum Mechanics* and models how materials behave under the action of forces. The most popular models used in this category are the isometric model [Chhatkuli et al., 2016, Vicente and Agapito, 2012, Bartoli and Collins, 2013, Chhatkuli et al., 2014a, Parashar et al., 2016, Chhatkuli et al., 2016] and the elastic (linear and non-linear) model [Agudo et al., 2012a, Agudo et al., 2012b, Agudo et al., 2016a]. The isometric model has been thoroughly studied and [Parashar et al., 2016] proved that isometric NRSfM is a well-posed problem. However, isometric priors are not accurate to describe deformations suffered by soft materials, such as a human organ. Elastic models have been proposed in [Agudo et al., 2012a, Agudo et al., 2012b, Agudo et al., 2016a], using Finite Element Method (FEM) approximations of linear and non-linear elastic materials. They estimate both camera pose and the 3D reconstruction of deformable objects from monocular scenes in real-time. These methods require object-dependent physical parameters, such as the Young’s modulus or its ratio with the Poisson coefficient. Besides, empirical evidence suggests that NRSfM is not well-posed with elastic constraints. It requires additional boundary conditions to limit possible ambiguities.

In *Statistical-based* approaches, the object’s shape space is assumed to be low-dimensional and is represented as the weighted sum of a set of basic shapes or shape basis. This idea was first proposed by [Bregler et al., 1999] and it is based on the low-rank assumption of the shape matrix. This low-rank prior has been studied by many researchers in the NRSfM literature [Del Bue et al., 2006, Torresani et al., 2008, Garg et al., 2013, Dai et al., 2012], as proved to be successful in many real world scenarios such as in the gestures in the human face. NRSfM based on the low-rank models is ill-posed, as the solution space is very ambiguous (several combinations of shapes, coefficients and camera poses could yield similar image projections). The efforts are thus concentrated in adding priors on the factorization of the tracking matrix. Some of them were estimating an initial rigid component [Del Bue et al., 2006], setting priors on temporal smoothness or spacial smoothness [Torresani et al., 2008], physical priors such as limiting stretching or extension in the surface [Vicente and Agapito, 2012, Brunet et al., 2010] or the use of trajectory bases [Akhter et al., 2009, Gotardo and Martinez, 2011] constraints for the point tracks along the sequence. Most of *statistical-based* approaches assume orthographic projection which allows to pose the problem as recovering a low-rank approximation of a matrix.

Other approaches, like [Fayad et al., 2010, Russell et al., 2011] tackled the problem in a piece-wise sense, using local models that better adjust certain parts of an object. The main drawbacks of these approaches are how to assign the initial partition set of models and how to assign the overlapping between points that share different models. This implies that global coherence of the model, even it could be better adjusted, is not guaranteed.

Most of aforementioned methods are based on tracking a sparse set of image correspondences, recovering a sparse 3D model of the object. As in SfM dense approaches have been recently investigated in NRSfM. The first one is [Garg et al., 2013] which performs dense optical flow combined with low-rank modeling and local smoothness priors. In [Russell et al., 2014] the segmentation and reconstruction of local rigid models is proposed. On the other hand, Kinect Fusion was further extended to handle deformable objects in Dynamic Fusion [Newcombe et al., 2015] for depth cameras.

## 1.4 Shape-from-Template (SfT)

A special case of deformable reconstruction is known as Shape from Template (SfT), where a reference model or template of the object is known and the objective is to find the deformed 3D shape given a single image or a sequence of images. This is known as model-based or template-based reconstruction. Most of SfT methods are based on *physics-based* deformation priors and in particular the isometric model [Brunet et al., 2010, Bartoli et al., 2015, Vicente and Agapito, 2012]. [Bartoli et al., 2015] describes the problem as a Partial Differential Equation (PDE) system and proves that imposing the isometry prior makes SfT a well-posed problem. Methods in SfT can be divided into local solutions, mainly based on solutions of a PDE system and global solutions, based on convex relaxations of isometry [Fua and Salzmann, 2011, Ngo et al., 2016].

The registration between the template and the model is needed in SfT. It is assumed known or manually done in many cases, although there are some automatic methods like [Pizarro and Bartoli, 2012].

## 1.5 Motivation

The pursue of this thesis is to propose a sequential solution to NRSfM that could run in real-time in a low-cost system based on CPU.

It has recently been shown the possibilities of applications based on rigid methods to be run efficiently with commodity software, i.e, not using GPU or other specialized hardware, with the goal of using them in embedded systems.

Taking as starting point PTAM, a well known state-of-the-art rigid SfM method and publicly available, this thesis tackles the improvements to be added (in both Tracking and Mapping threads) in order to deal with deformable objects.

Another point that this thesis tackles is assuming perspective projection instead of orthographic projection, assumed by most of NRSfM methods. The perspective projection is an accurate projection model and does not suffer from many depth ambiguities existing in the orthographic model. Adapting this model in NRSfM represents another challenge.



Most of the current state of the art algorithms rely on the assumption of not actually computing the tracking and data association, whereas in a real application this is one of the most important challenges to face. In this thesis these two tasks will be conveniently studied.

## 1.6 Applications

Several applications of SfM and NRSfM are summarized in the following sections.

### 1.6.1 Augmented reality, apps and games

The potential of the SfM and NRSfM on AR, mobile apps and games has started to be exploited in the last years. AR needs SfM for localization and mapping of objects where virtual objects are visualized in real-time.

With respect to the AR, there are libraries specifically designed to help with this task. One of the most representative ones is Wikitude, although the most famous one on the market is Vuforia, which was initially developed by Qualcomm as a trial to diversify its main business line, but it was sold to the PTC company. Now, this library is widely extended and can be used by several platforms: Android, IOS, Unity3D and recently Hololens and Windows 10. As an example, there are computer games implementing AR in some extend, like role card games. An example is given in Fig. 1.1 which also includes an screenshot of Vuforia working on Unity3D and a live camera.

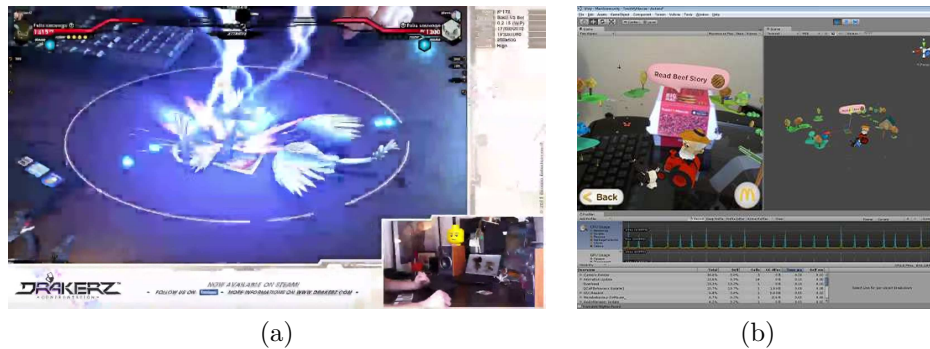


Figure 1.1: (a) An example of a card game using AR (Drakerz). (b) An example of the required libraries for Unity3D Vuforia

There are several modules involved on a simplified AR pipeline: acquisition / sensing, processing and representation.

Regarding the acquisition modules there are peripherals that help on the sensing (accelerometers and other sensors) or to perform the reconstruction. The breakthrough was

the Kinect sensor in their two versions: Structured Light and Time of Flight (ToF). Similar versions of this sensor is available by other brands like Asus, Intel, etc. An example of them could be seen in Fig. 1.2, as well as an example of the skeleton tracking directly available on the drivers.



Figure 1.2: (a) Kinect v1 device, (b) Kinect Skeleton Tracking example.

Rendering devices can be divided into two types: glasses and standalone. In the first category we can find the Oculus Rift, HTC Vive and lately PS Virtual Reality (VR). In the second category we can find devices like Hololens (MS) and Magic Leap (a Google Venture). In Fig. 1.3 the Hololens device is seen jointly with a proof-of-concept example of the Magic Leap technology.

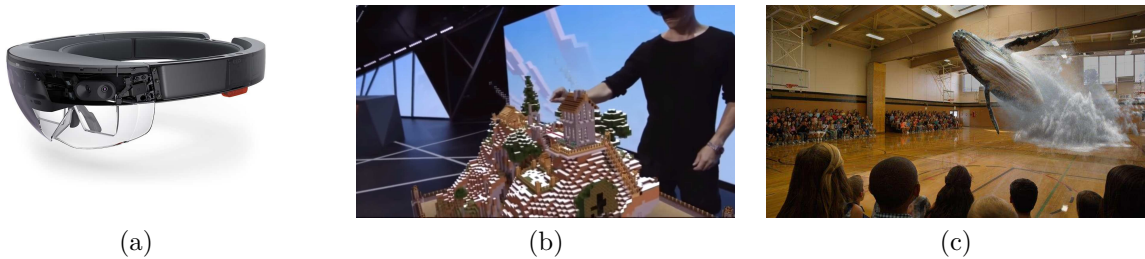


Figure 1.3: AR / VR applications. (a) Hololens device. (b) Hololens running with a Minecraft demo from the virtual camera view. (c) Magic Leap with a whale getting out of the water

Mobile devices can also be used to apply AR as it has been demonstrated in recent apps as Pokemon Go game in its AR mode (Fig. 1.4.a). There are apps performing reconstruction of an object from several views either online or offline. There is an interesting example of an app developed by Disney co. with live re-texturing of a 3D character using NRSfM when the page is turned [Magenat et al., 2015] (Fig. 1.4.b). PS Vita has also some games implementing AR, as it can be seen in Fig. 1.5.

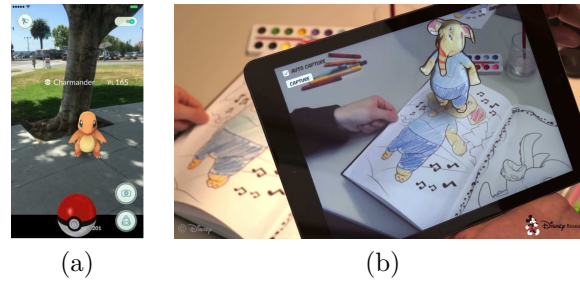


Figure 1.4: Apps using SfM: (a) Pokemon Go / NRSfM: (b) Disney Research to repaint 3D characters live



Figure 1.5: AR on PSVita games. (a) Invizimals. (b) PulzAR

### 1.6.2 Film industry / Cinema

Motion capture technique is massively used in the FX (special effects) industry. Many approaches are based on point markers. An example can be seen in Fig. 1.6.a. The “Virtual Camera”, as seen in “Avatar” movie, helps the directors and camera crew with a preview of the scene that can be easily rendered, so as to guide the director in a closer way to the final result, as it can be seen on Fig. 1.6.b.

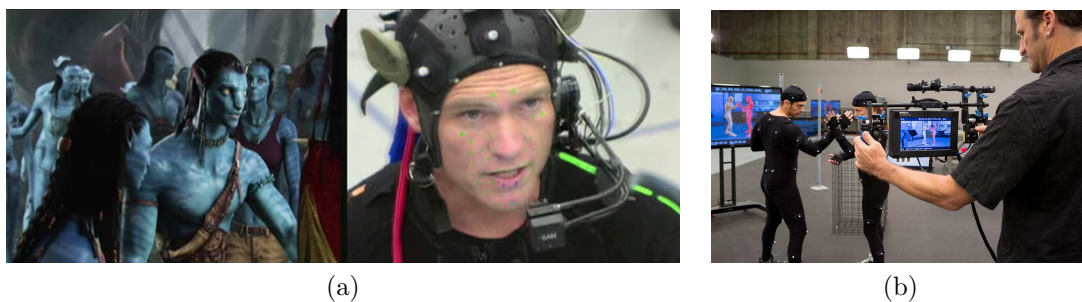


Figure 1.6: (a) Motion Capture systems. (b) Virtual camera systems. This is extensively used on the film industry to add the special effects.

### 1.6.3 Marketing

There are also many applications in the field of marketing where AR has been used as a novel asset. Good examples are the virtual fitting room made by Toshiba (Fig. 1.7.a) and the so-called N-show by New Tempo (a Chinese company) in which you could try clothes

or complements before actually buying them. There is also a virtual makeup mirror based on Kinect that simulates makeup in real-time.

Additionally, some marketing strategies are based on cards for kids including some AR games (Fig. 1.7.b). Clothes can also contain patterns on them that the apps are able to detect. By taking a video with a phone, an animation is shown to the user. An example can be seen in Fig. 1.7.c.

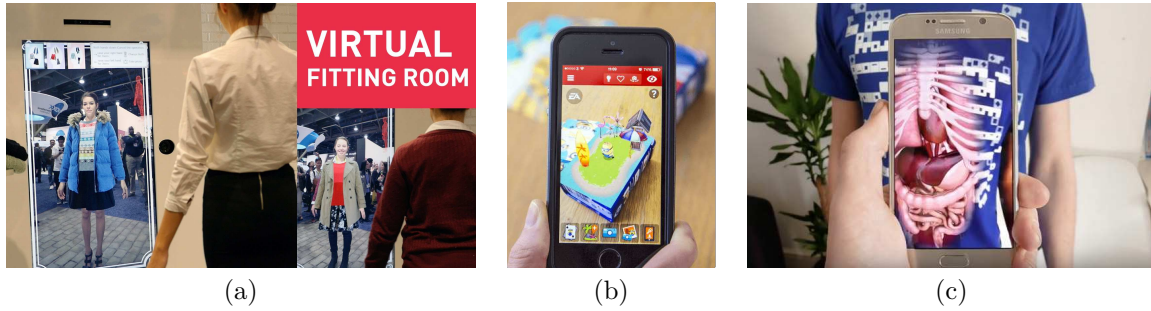


Figure 1.7: Marketing examples: (a) an example of a virtual fitting room. (b) an AR minion App used for a campaign for a supermarket. Screenshot taken from zappar web. (c) a shirt with visual markers which are recognized by an app and represents an animation taking into account the camera pose.

#### 1.6.4 Surgery training / simulation. Medical imaging

AR is a well suited technique to help surgeons planning surgery and is a valuable tool for training. An example of a surgery simulator can be seen in Fig. 1.8.a.

In medical applications, NRSfM is a suitable technique to be applied as human organs and tissues are deformable objects. An extensive study of the state of the art with respect to the reconstruction based on laparoscopic images is shown in [Maier-Hein et al., 2014]. An example of a reconstruction based on an angiography is shown in Fig. 1.8.b.

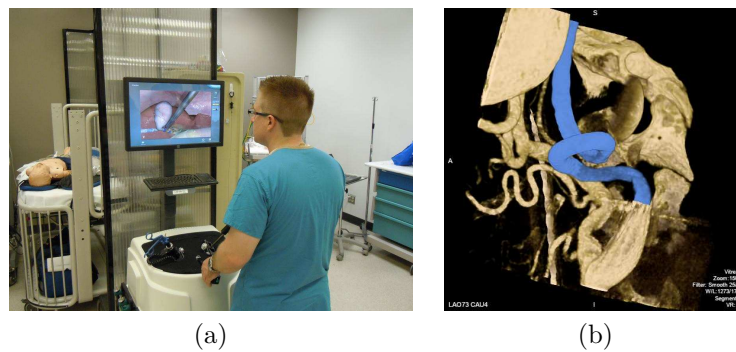


Figure 1.8: Situations in which NRSfM could be applied. (a) A virtual simulation trainer, at the center a live 3D reconstruction. (b) The 3D reconstruction of an angiography.

### 1.6.5 Material research / Modelling

A way to compute deformation parameters along more than one dimension on live sequences can be to record the sequence of objects with visual markers placed on them. It can also be done using a standard registration system and then to compute the sequence offline, applying NRSfM techniques. There are also works for modeling a sailing boat flag to check how well could the competitors be based on the shape of the sailing boats.

### 1.6.6 Arts

There are some artists using the latest VR / AR technologies to get another way of artistic expression. An example is provided by Google artists using VR glasses while doing different traces in a virtual 3D environment. This can be seen on [virtualart.chromeexperiments.com](http://virtualart.chromeexperiments.com). This is also applicable to insert art elements on the real world and load them using the adequate app or visor, as it is already done for marketing apps.



## Chapter 2

# Related work

The process of reconstructing a 3D structure, either a set of sparse points or a dense mesh or surface, from 2D images is a fundamental problem in computer vision. Without any other precondition this problem is ill-posed. The most popular prior used for reconstruction is based on scene's rigidity. Image flow is then caused by camera motion and used as main visual cue. Jointly estimating 3D shape and camera motion from a set of images is known as Structure from Motion (SfM). When the position of several cameras is known and a several cameras capture the scene simultaneously the setup is known as Multi View Stereo (MVS). In this setup 3D is directly estimated by triangulation methods. Many works in the literature use other visual cues and priors to recover 3D from images, such as focus, shading or lighting. These techniques are out of the scope of this thesis and for more information we refer the readers to [\[Paladini, 2011\]](#).

Other reconstruction methods are based on the use of specific sensing hardware such as laser scanners or Red Green Blue and Depth (RGBD) sensors, like Time of Flight (ToF) cameras or sensors based on the projection of structured light, such as the popular Kinect v1 sensor.

The working principle of structured light is as follows: an image with a certain pattern is projected onto the scene and its projection in the image allows one to recover the 3D geometry. In order to hide the pattern image in commercial systems and to avoid inconveniences, the Infra Red (IR) spectrum is commonly used for these purposes.

With respect to the ToF cameras they use modulated IR emitter to illuminate the scene. Using fast per-pixel correlation methods, depth is obtained from the phase change measured at each pixel. An extensive explanation of the ToF principle can be found in [\[Sarbolandi et al., 2015\]](#).

These active sensors are accurate but more expensive than passive methods based on cameras. They are also more difficult to be embedded on a portable device such as a smartphone or to be used in laparoscopy cameras. This thesis focuses on monocular 3D reconstruction from images.



## 2.1 Structure from Motion (SfM)

We refer as SfM to the problem of inferring the 3D geometry of a rigid scene given a set of images where the camera moves relatively to the scene. The rigidity assumption makes SfM a well-posed problem and it fits with many real applications. For instance, the geometry of static objects like houses, streets, mountains, walls, doors, furniture, etc. hardly ever changes, at least during the scene recording.

There are two main approaches for rigid reconstructions: *batch* and *sequential* approaches. In batch methods all the information of the sequence is available at processing time and thus reconstruction is performed offline. These methods are mainly based on Bundle Adjustment (BA) and use all constraints and image measurements at the same time. This yields very accurate solutions. On the contrary, in the sequential/online approach the 3D structure and camera pose are updated with each new frame available. This usually produces less accurate solutions with respect to batch methods but it admits real time applications.

The problem of sequential localization and map generation is known as Simultaneous Localization And Mapping (SLAM) in robotics. It has been studied for a large variety of sensors, such as cameras, ultrasound, Light Detection and Ranging (LIDAR), laser, etc. Vision-based SLAM is now widely used in robotics, showing that a camera mounted on the robot can be used to recover both the robot pose and a map of visual features. This represents a key technology for robot navigation as well.

### 2.1.1 Sparse reconstruction

The foundations of SfM are described in [Hartley and Zisserman, 2004]. The geometry of multiple views of a rigid scene is well known and links camera motion with image measurements. It generalizes to other reconstruction problems, including multiple camera setups, where their position is known. The reconstruction procedure in these cases could be materialized via triangulations, tensors, factorizations, etc, and further optimized using some algorithms such as Gauss-Newton, gradient descent, Levenberg-Marquard (LM), etc.

#### 2.1.1.1 Batch approaches

Early SfM approaches are based on matrix factorization. This technique decouples the camera pose and map estimation and usually assume the orthographic camera. The first important attempt was proposed in Tomasi and Kanade work [Tomasi and Kanade, 1992]. Matrix factorization was extended later to the perspective camera by Sturm and Triggs in [Sturm and Triggs, 1996] using an iterative alternation approach. Factorization methods are not very accurate and nowadays are used as a way to initialize refinement methods based on BA [Triggs et al., 1999]. This algorithm is based on the minimization



of the reprojection error of all 3D features (points and line segments) seen in the images. This gives a maximum likelihood estimate of the parameters (pose and 3D shape) when the image noise is assumed to have a Gaussian distribution. BA is commonly used in large scale batch problems where the processing time is not an issue. The most famous example is in [Agarwal et al., 2009], where the reconstruction from a collection of Internet photos of the city of Rome is proposed. BA implies the optimization of non-convex functions. Iterative solvers are usually employed, such as Gauss-Newton or LM [Hartley and Zisserman, 2004]. The Hessian matrix required by these methods grows quadratically with the number of image measurements. In many problems they are usually higher than ten or hundreds of thousands, and then the solution of the normal equations requires exploiting the sparse structure of the Hessian matrix. A good initialization is crucial in BA methods. Otherwise the optimizer is likely to fall into a sub-optimal local minimum.

Some aspects of BA are improved in recent works. Regarding the initialization of BA, [Gong et al., 2015] use boundary constraints for certain parameters in the reconstruction that improve global optimality of BA. The work of [Cui and Tan, 2015] presents a solution for the case in which there are degenerate camera configurations, extracting a basic depth map and then giving an approximation of the camera pose for the BA algorithm.

Regarding how the different images are processed in BA, [Schonberger and Frahm, 2016] propose an incremental SfM framework and introduced pre-triangulations for incremental reconstructions prior to the execution of the BA. The proposal of [Cohen et al., 2015] considers a combinational approach to solve loop closures without temporal information. It also use sub-models, individual components and symmetry priors to solve ambiguities. In the work of [Eriksson et al., 2016] the BA algorithm is reformulated to be applied in a distributed manner, which allows the implementation on cloud systems.

### 2.1.1.2 Sequential approaches

Sequential SfM is important in many applications. In robotics, sequential SfM is known as visual SLAM. It was first proposed as a sequential Bayesian inference problem in [Davison et al., 2007] using the Extended Kalman Filter (EKF). In Smoothing And Mapping (SAM), the problem is seen as a graph reduction problem, as depicted in [Dellaert and Kaess, 2006].

In [Klein and Murray, 2007], Parallel Tracking And Mapping (PTAM) was proposed as a solution to visual SLAM. This was considered a breakthrough in SLAM methods, capable of giving enough quality for Augmented Reality (AR) applications. One of the main reasons of the success of PTAM was its system architecture. The processing paradigm changed from pure sequential to parallel. PTAM remarkably improved stability and reconstruction accuracy over previous approaches. It's also able to handle bigger maps in real time with low cost hardware. This Thesis follows the PTAM philosophy and thus more details about this methods will be given in section 2.1.2.

PTAM is considered a solid reference on the state-of-the-art for SLAM systems and many methods are inspired on it [Pan et al., 2009, Newcombe and Davison, 2010, Stühmer et al., 2010, Tan et al., 2013] (a detailed explanation of each one can be found in section 2.1.2). In [Stühmer et al., 2010] and [Newcombe and Davison, 2010] authors moved towards dense reconstruction, using the estimations of PTAM as base of their posterior estimations. Dense approaches taking only images or RGBD data will be explained in section 2.1.3. In addition [Klein and Murray, 2009] showed that PTAM could be run in portable devices.

Sequential and sparse SfM methods are suitable now to be run in real time in portable devices. [Wang et al., 2015] provides an example of a mobile implementation of a mall indoor localization based on visual marks, not strictly on feature points but textual marks and edges. In addition, it uses the mall map as a source of information to get the whole map in advance. It roughly estimates the width of each of the shops on the mall and tries to read the text from the shop panels. With those clues it tries to match with the information provided with the given map so as to give an approximated indoor localization.

With respect to commercial computer visions systems for smartphones, there are two main trends in the market:

- On the one hand, the whole processing is done on the device, as it was indicated in the previous examples. This approach is sometimes restricted to certain types of devices with enough resources to afford the computation. In addition, this approach is one of the most CPU consuming ones.
- On the other hand, images are acquired, transmitted, processed "on the cloud", and then the results are sent back to the device. This adds network costs but lowers battery consumption and processing resources in the phone. The general quality, bandwidth and user experience of the application depends on the mobile network the device is connected to (2G / 3G / 4G / 5G).

Some AR companies like Wikitude supports both working modes (device and cloud) whereas there are applications that only work on cloud or on device.

### 2.1.2 Parallel Tracking and Mapping (PTAM)

As was mentioned before, PTAM is considered a breakthrough in visual SLAM. This method was first introduced in [Klein, 2006, Klein and Murray, 2007], further improved in [Klein and Murray, 2008] by adding edges to the tracking system. It was optimized to be implemented in an iPhone device in [Klein and Murray, 2009].

PTAM was also taken as a base for other projects like [Pan et al., 2009, Newcombe and Davison, 2010, Stühmer et al., 2010, Tan et al., 2013], as it provides a good setup to get a set of sparse point tracks and a stable camera pose.

PTAM separates sequential SfM into two parallel process: the *tracking thread* and the *mapping thread*. This multi-threading architecture is compatible with multicore processors, available for consumer electronics and portable devices. As mentioned in [Newcombe et al., 2011a], this has several advantages over SLAM methods based on statistical filtering, where the size of the state vector (camera pose and map points) becomes huge over time and they suffer from accumulative drift error. Splitting the mapping thread and the tracking thread in PTAM was an advantageous alternative to the full propagation of the state vector frame by frame. It reduced drift and increased the amount of map points by an order of magnitude that could be handled in real time compared to its predecessors. We describe next the PTAM algorithm.

### 2.1.2.1 The tracking thread

The tracking thread is in charge of the following tasks:

1. When there is no map to track, it starts tracking points from the incoming images, interacting with the user to select the first two representative keyframes. When the user finishes and there are enough tracks, the track pairs are sent to the mapping thread to create the map using a camera stereo approximation. A simple correlation matching is followed in this case. Only the most robust matches are matched using this mechanism. The tracking thread is only responsible of providing good tracks to the mapping thread in order to generate the initial map based on them.
2. When there is a map available, the tracking thread behaves different, as there are map points to track over the frames. In this case, the process is the following:
  - First, the images are converted to grayscale, stored and downsampled 3 times to create a 4 pyramid level image. On this pyramid, FAST features [Rosten and Drummond, 2006] are extracted without maximal suppression. With this, a handcrafted set of fast multilevel features is designed.
  - When a frame is acquired, a prior estimation of the pose is generated from the motion model. The motion model consists of a decaying velocity model that only takes the velocity of the current and the previous frames:

$$v = \alpha v_{current} + (1 - \alpha) v_{old} \quad (2.1)$$

where  $v$  represents the estimated velocity,  $v_{current}$  is the currently estimated velocity from the image,  $v_{old}$  is the previous estimated velocity and  $\alpha$  represents the parameter controlling the decay of the model.

The motion model is applied on the pose as a regular update on the  $\mathbb{SE}(3)$  Lie group space (briefly explained in [Klein, 2006]) with the camera velocity already computed as:

$$E'_{CW} = \exp(v) E_{CW} \quad (2.2)$$

where  $E_{CW}$  denotes the rigid transformation matrix from world to camera coordinates. The velocity of each frame is estimated using the properties from the  $\mathbb{SE}(3)$  group. If blurring rotation is activated, the velocity is computed with the help of [Benhimane and Malis, 2007].

- Map points are projected according to the previous pose estimation using the perspective projection model including barrel radial distortion correction:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \frac{r'}{r} \begin{pmatrix} \frac{x}{z} \\ \frac{y}{z} \end{pmatrix} \quad (2.3)$$

with the correction terms  $r$  and  $r'$  calculated as follows:

$$r = \sqrt{\frac{x^2 + y^2}{z^2}} \quad (2.4)$$

$$r' = \frac{1}{\omega} \arctan \left( 2r \tan \frac{\omega}{2} \right) \quad (2.5)$$

where  $(x, y, z)$  are the three spacial coordinates of the points in camera coordinates,  $f_u, f_v$  are the focal distances,  $u_0, v_0$  is the projection center of the camera and  $\omega$  is a radial distortion parameter computed during camera calibration.

- Before continuing with the rest of the steps, using just the previous estimation of the pose and the current frame, the points are searched on the current frame. A fixed range search based on an affine warp on each point is performed around the last found localization of the patch. An affine warp is computed for each point. This warp is based on the displacement of the point on the current level w.r.t the base level of the pyramid.

The warp is found by projecting pixel displacements in the source keyframe onto the patch's plane, and then, on the current frame. It gives an idea whether the current pyramid level is correct, corrections in perspective, etc. The patch intensities are normalized to reduce appearance changes due to illumination variations. At this stage, the point can be marked as not found, so it can be discarded for some of the computations described below.

- Coarse-scale features are searched in the image and then a coarse estimation of the pose from the previous estimation is obtained.
- Then, the fine-scale features are re-projected and scaled in the image. Camera pose is then updated using the error of those fine scale selected candidates from the total amount of points. A Gauss-Newton optimization method is used. The maximum number of iterations is 10. Depending on the number of the iterations

already performed for each frame, the treatment of outliers can be more or less restrictive.

- The camera update is found by minimizing the following robust objective function:

$$\mu' = \arg \min_{\mu} \sum_{j \in S} \text{Obj} \left( \frac{|e_j|}{\sigma_j}, \sigma_T \right) \quad (2.6)$$

where  $\text{Obj}$  is the Tukey bi-weight objective function [Tukey, 1960].  $\mu$  is the state vector of the 6 degrees of freedom formed by the 3 rotation angles and the 3 translation axes,  $e_j$  is the error of the current sample,  $\sigma_j$  is the standard deviation of the current error distribution and  $\sigma_T$  is the standard deviation threshold of the objective function. The implementation of the camera update estimation is done using Weighted Least Squares (WLS) and each of the weights is given by the M-estimator.

- Once the update estate  $\mu'$  is computed, it is applied to the pose following the rules of the  $\mathbb{SE}(3)$  group:

$$E'_{CW} = \exp(\mu') E_{CW} \quad (2.7)$$

- Quality is measured at every frame as a fraction of feature observations which have been successful.
  - If it is considered poor, tracking continues normal but no more keyframes are added to the mapping thread.
  - If it is very poor during some frames, the tracking is considered lost and a recovery procedure is initiated.

### 2.1.2.2 The mapping thread

This thread is in charge of creating and maintaining the map. Previous approaches considered the map as the set of 3D points. In this approach, the map was extended:

- The map contains the set of keyframes, taken at certain times, that include the camera pose and the features seen in the keyframe.
- Each point feature represents a locally planar textured patch in the world and it contains an estimated unit patch normal to the textured patch (for the use of the warping matrices).
- Each of the keyframes stores a 4 pyramid level of gray-scale images.
- Each map point stores a reference to the first keyframe that was detected and the pixel location for each pyramid level. It also stores the computed 3D position on world coordinates.

- Normal patch size is 8x8, but it depends on:
  - The pyramid level.
  - Distance from the source keyframe camera center.
  - Orientation of the patch normal.

The main difference corresponds to the keyframes used to sample the scene at a certain time. This change allows a severe saving in memory and CPU, as not all the sequence frames must be processed.

Processing only keyframes in the mapping thread reduces real-time restrictions present in the tracking thread. The whole map can be obtained from the keyframes with accurate refinement methods such as BA. Features are re-visited and full map optimizations are performed as new keyframes are included in the map.

The mapping tasks must be performed in a separate thread, as they are CPU and memory intensive. Since the mapping is separated from the tracking, the mapping thread can concentrate on reducing the overall error running BA.

In the case of hand-held camera, and because of the movements, data association errors become a problem and the generated maps could be corrupted. To solve this problem it is necessary to use outlier rejection algorithms.

Fig. 2.1 depicts the flowchart of the mapping algorithm. Hereafter we discuss the most important points of the mapping steps:

1. The initialization mechanism is based on triangulation. It searches correspondences along the epipolar line. RANSAC (RANdom Sample Consensus) is proposed to discard outliers [Stewenius et al., 2006]. The whole initial map estimation algorithm assumes a distance between initial keyframes of approximately 10 cm to start constructing the map, as with a monocular system any reconstruction is always provided up to a scale factor. This initialization procedure does not allow rotations, only pure sideways translations. Afterwards BA is run on this map to refine the initialization. Finally, the map is aligned to the plane  $z = 0$ .
2. Once the map is constructed from the first two keyframes, this thread is also in charge of checking the map growing, updating it by using keyframe distance criteria. The map is expanded if unknown regions are explored and known features from other keyframes can be seen. There are some restrictions to add a keyframe to the map:
  - Good tracking quality (measured by the tracking thread)
  - Minimum time of 20 frames between keyframes
  - Minimum distance from the nearest point already included in the map, to avoid corruption and ensuring enough baseline to estimate new 3D points.

3. The sparse BA implemented in PTAM is the one proposed in [Hartley and Zisserman, 2004], where several variants of iterative estimation methods can be found. It implements the algorithm A6.4 of the book which describes the general sparse LM algorithm. It does not take into account the Hessians which speeds up the computation and has a negligible impact in accuracy.
4. This thread also handles outlier control, accounting, recycling and discarding points if necessary. Recycling means that some points are given a second opportunity if they are found on other frames and are further refined.
5. Over time, the map could be big enough to be efficiently handled with a regular BA implementation. Instead, a local area BA is run for a neighborhood of incoming keyframes and then, when there is enough time, it is run on the global map. Local BA reduces computational cost from  $O(N^2M)$  to  $O(NM)$

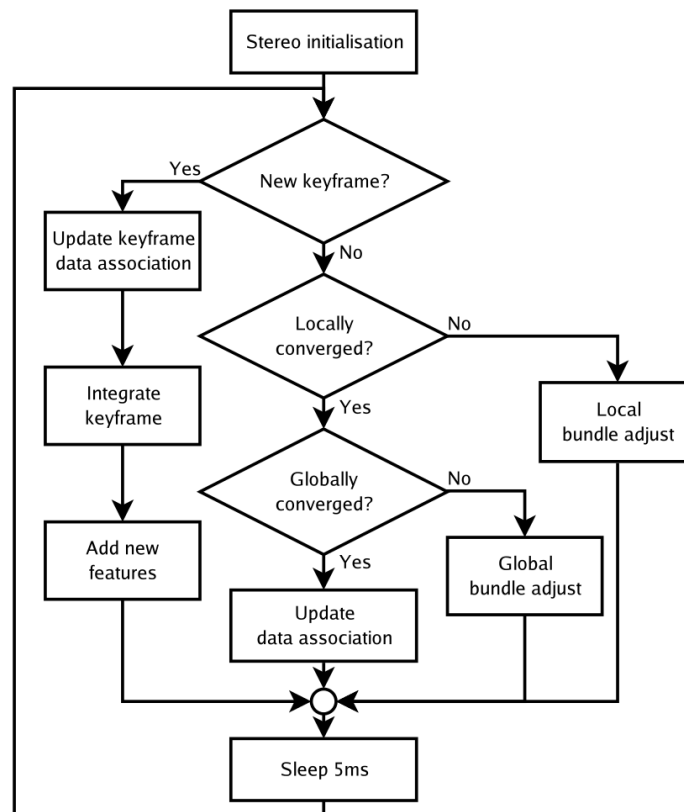


Figure 2.1: PTAM mapping workflow diagram. Taken from [Klein and Murray, 2007]

An example of the live mapping operation can be seen in Fig. 2.2.

Compared to PTAM, Davison's EKF-SLAM [Davison et al., 2007] has three main weak points: 1) it needs a smooth camera movement prior to correctly estimate depths; 2) it



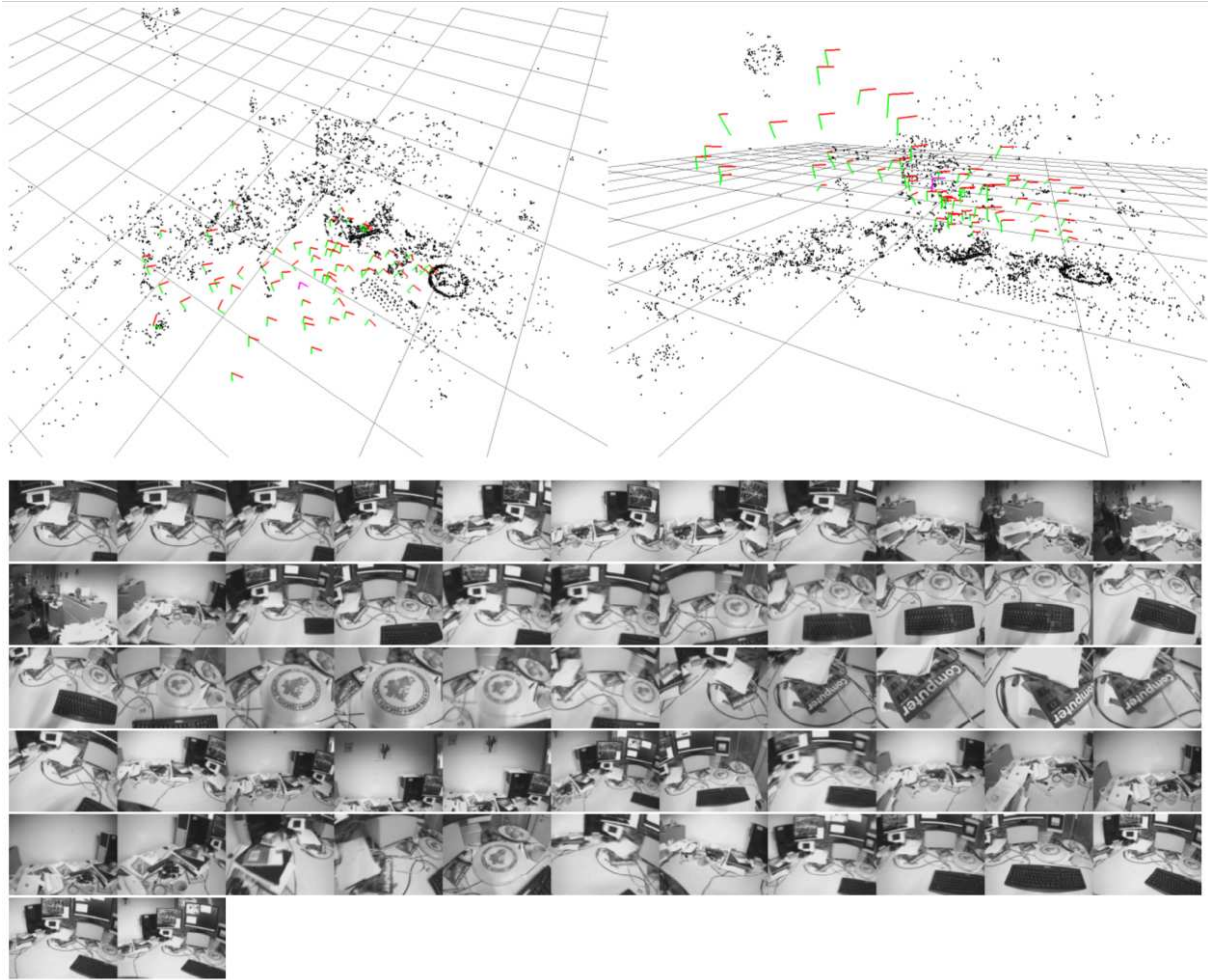


Figure 2.2: PTAM incremental mapping example. Extracted from [Klein and Murray, 2007]

has small tolerance to motion blur and variation in focus and 3) it could handle less points than PTAM.

This comparison can be seen in Fig. 2.3.

However, PTAM has also some drawbacks:

- It generally fails with severe blurring. This was mitigated by adding edges on the tracking [Klein and Murray, 2008].
- It is not robust to repetitive structures (due to the use of a robust estimator), as it was not designed for outdoors but for little workspaces. Trying to add a keyframe when there is a repetitive structure could yield to a wrong estimation of the keyframe pose. This would corrupt the map by insertion of the new keyframe with an erroneous pose.
- Partial occlusions could affect the way that the points are inserted into the map. This problem is solved in dense approaches [Newcombe and Davison, 2010, Newcombe et al., 2011b]. As the model is sparse, no occlusion reasoning is possible.



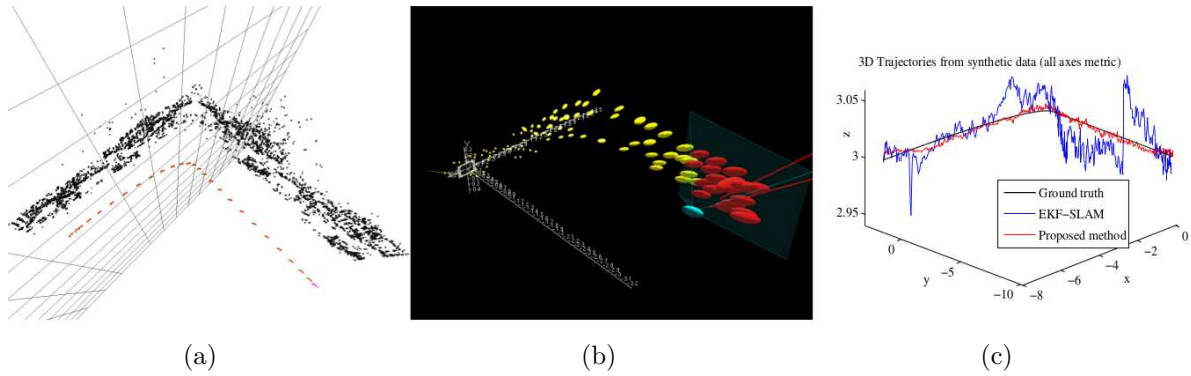


Figure 2.3: EKF-SLAM vs PTAM comparison. (a) shows the map produced by PTAM. (b) shows the map generated by the latest version of EKF-SLAM at that date. (c) shows the trajectories compared with the ground truth. Extracted from [Klein and Murray, 2007]

The use of PTAM with deformable objects has the following undesirable effects:

- The initialization depends on having a pure translation and a perfect rigid environment.
- In the case a point is deforming, the BA could automatically discard it by marking it as non-visible several times or considering it as an outlier.
- The tracking does not allow deformations on the points, as it is an standard SfM system. The deformation would be approximated to the closest rigid movement possible from the given map.

### 2.1.3 Dense reconstruction

Dense reconstructions involve thousands of points for each image. These reconstructions are mainly based on Total Variation (TV) algorithm [Curless and Levoy, 1996]. The main advantage of these systems with respect to the sparse feature-based methods is that they are less limited by the texture contents of the objects. They estimate the variation between frames using optical flow.

The first approaches able to build dense reconstructions were built on top of sparse reconstructions, such as [Stühmer et al., 2010] and [Newcombe and Davison, 2010]. Both methods used PTAM for obtaining poses and interest points estimations in the initialization.

Dense Tracking And Mapping (DTAM) [Newcombe et al., 2011b] got rid off the initial pose estimations given by PTAM, as they got full dense mapping and tracking based on the volumetric data. The tracking was performed by full image alignment. The improvement reached with this change can be seen in Fig. 2.4.

The mapping thread is continuously updating a dense depth map which is globally optimized with TV with L2 norm and iteratively smoothed. The algorithm can be run in real time with the help of an efficient implementation in GPU. An example of the depth maps computed for the map generation and a comparison with the PTAM features is shown in Fig. 2.5.

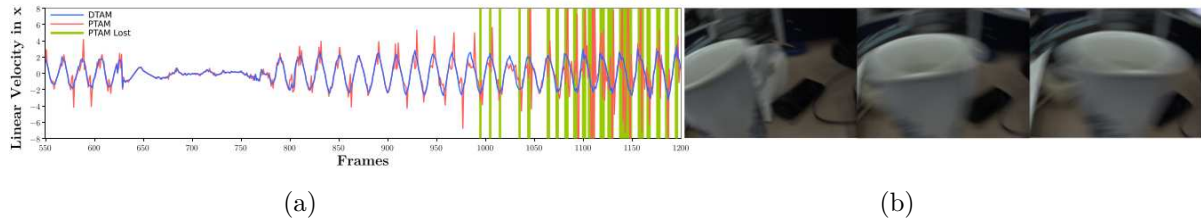


Figure 2.4: DTAM vs PTAM tracking comparison for fast movements. (a) Linear velocities for DTAM (blue) and PTAM (red) over a challenging high acceleration back-and-forth trajectory close to a cup.

PTAM tracking losses are shown in green. DTAM's linear velocity plot reflects smoother motion estimation. (b) shows samples from the sequence. Extracted from [Newcombe et al., 2011b]

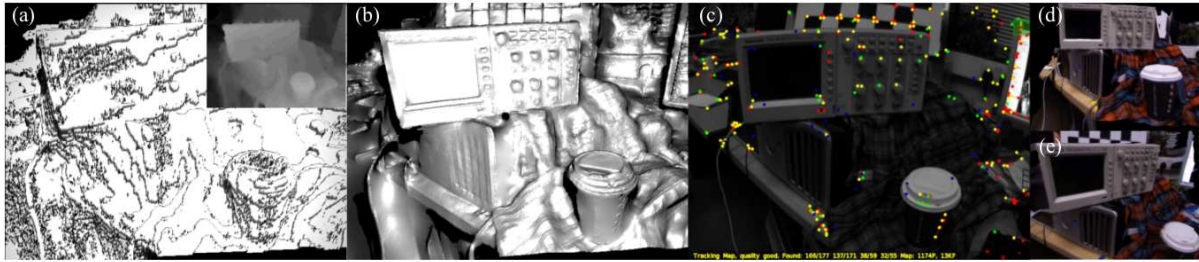


Figure 2.5: DTAM vs PTAM comparison. Inverse depth maps in (a) without subsample refinement and in (b) with subsample refinement. Same scene in (c) with PTAM. (d) and (e) Novel wide baseline mapped views of the reconstructed scene used for tracking in DTAM. Extracted from [Newcombe et al., 2011b]

The way in which the camera pose is tracked in MonoFusion [Pradeep et al., 2013], is similar to PTAM except for some modifications. It proposes a similar system to DTAM but relaxing the expensive pose estimation given by the Total Variation algorithm applied on the whole image. It could be seen as a similar approach as the presented in [Newcombe and Davison, 2010].

With the appearance of the Kinect sensor, and having the drivers publicly available, the RGBD data can be used and fused with a similar volumetric approach as DTAM, used in Kinect Fusion [Newcombe et al., 2011a]. It aligns the depths maps given by the Kinect sensor instead of computing them from the images by a multilevel Iterative Closest Point (ICP) alignment algorithm. Some reconstruction examples could be seen in Fig. 2.6.

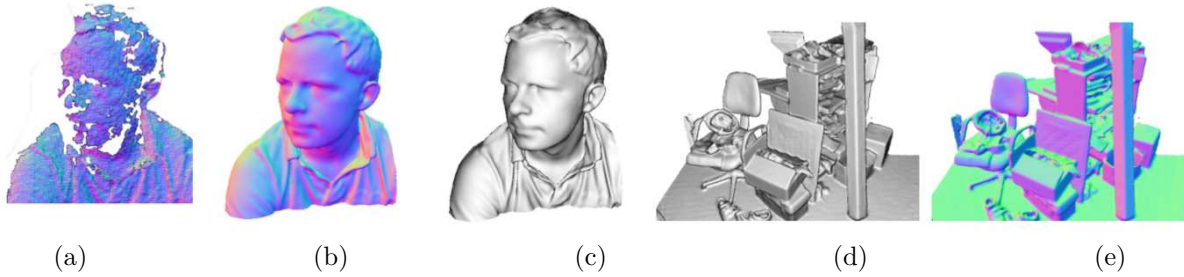


Figure 2.6: Kinect Fusion examples. Generated in real time with a handheld Kinect sensor. (a) A noisy and incomplete data from the Kinect sensor, (b) the normal maps, (c) the 3D model including shading. (d) another example of a desktop and (e) its normal maps. Extracted from [Newcombe et al., 2011a]

There is an experimental open source version available on PCL library [Rusu and Cousins, 2011], in which a tutorial for large scale reconstruction to generate a textured mesh for the Kinect captures is included. There is also an open source project that uses only CPU to produce volumetric maps called FastFusion [Steinbruecker et al., 2014].

As more CPUs and reduced GPUs are integrated on the mobile phones, dense SfM has become a recent reality. In [Schöps et al., 2014] a semi-dense approach in which intensities values are matched between images is presented. In [Ondruska et al., 2015] they present a system for real-time dense 3D reconstruction running in a smartphone. It is based on selecting keyframes where the depth map is estimated. The dense volumetric 3D model is then fused with the IMU information to solve dense image alignment and pose estimation. Still, these dense approaches are still only possible in high-end devices. Sparse methods are thus preferred in mobile platforms.

A taxonomy of the presented approaches is depicted in Table 2.1.

## 2.2 Non-Rigid Reconstructions

So far, the presented approaches are able to reconstruct scenes that are rigid, where the geometry does not change over time and the image motion is due to the relative motion between the camera and the scene.

With deformable objects the rigidity prior cannot be applied and thus classic SfM methods fail. In order to make the problem solvable, deformation priors are introduced, as it is shown in 2.2.1. Section 2.2.2 shows a summary of the different model-based reconstruction proposals. Model-free or Non-Rigid Structure from Motion (NRSfM) is explained in Section 2.2.3.

### 2.2.1 Study of the ambiguities and deformation priors

The solution space of non-rigid reconstructions is much bigger than when rigidity is imposed. It is not only constrained by the 6 Degrees of Freedom (DoF) of rigid space

Rigid approaches	sparse/dense	GPU	RGB-D	mobile	sequential	real-time
[Tomasi and Kanade, 1992] (factor)	sparse	no	no	no	no	no
[Sturm and Triggs, 1996] (factor)	sparse	no	no	no	no	no
[Tang and Hung, 2002] (factor)	sparse	no	no	no	no	no
[Dellaert and Kaess, 2006] (SAM)	sparse	no	no	no	yes	yes
[Davison et al., 2007] (mono-SLAM)	sparse	no	no	no	yes	yes
[Klein and Murray, 2007] (PTAM)	sparse	no	no	no	yes	yes
[Klein and Murray, 2009] (PTAM)	sparse	no	no	yes	yes	yes
[Agarwal et al., 2009](BA)	sparse	no	no	no	no	no
[Pan et al., 2009]	sparse	no	no	no	yes	yes
[Tan et al., 2013] (PTAM-like+SIFT)	sparse	yes (SIFT)	no	no	yes	yes
[Gong et al., 2015] (BA)	sparse	no	no	no	no	no
[Cui and Tan, 2015](BA)	sparse	no	no	no	yes	no
[Cohen et al., 2015](BA)	sparse	no	no	no	yes	no
[Schonberger and Frahm, 2016] (BA)	sparse	no	no	no	yes	no
[Eriksson et al., 2016] (BA,cloud)	sparse	no	no	no	no	no
[Stühmer et al., 2010] (PTAM+TV)	dense	yes	no	no	yes	yes
[Newcombe and Davison, 2010] (PTAM+TV)	dense	yes	no	no	yes	yes
[Newcombe et al., 2011b] (DTAM)	dense	yes	no	no	yes	yes
[Newcombe et al., 2011a] Kinect Fusion	dense	yes	yes	no	yes	yes
[Pradeep et al., 2013] Mono Fusion	dense	yes	no	no	yes	yes
[Steinbruecker et al., 2014] Fast Fusion	dense	no	yes	no	yes	no
[Schöps et al., 2014] (PTAM-like+inv depth map)	semidense	no	no	yes	yes	yes
[Ondruska et al., 2015] Mobile Fusion	dense	yes	no	yes	yes	yes

Table 2.1: Rigid SfM approaches summary

transformations, but it is also affected by the DoFs imposed by the possible deformation on the object. In order to better illustrate the problem, we can see the example of Fig. 2.7.

Deformation priors are necessary to properly constrain any deformable reconstruction problem. Existing deformation priors can be divided into two main groups: *i) Physics-based* and *ii) Statistical-based* models.

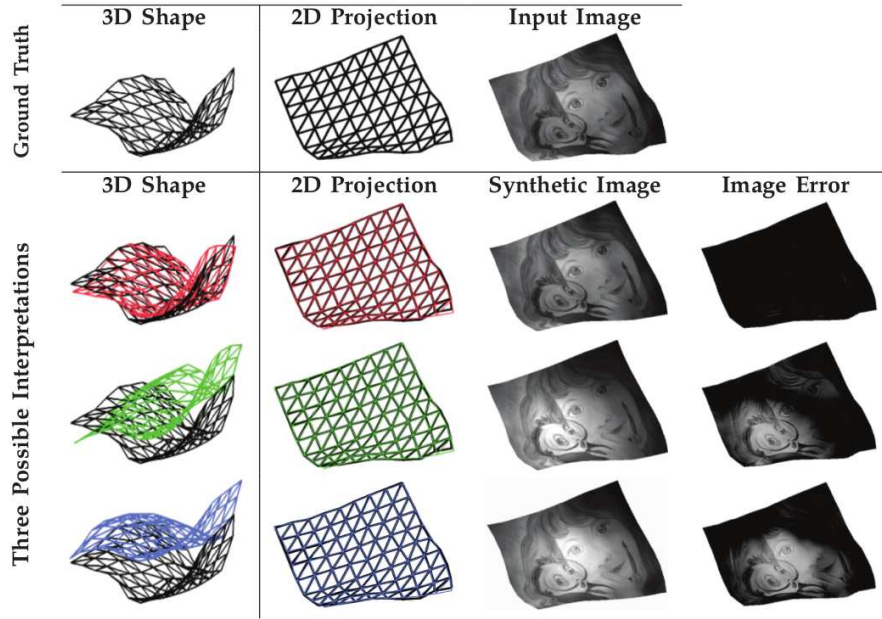


Figure 2.7: Ambiguity illustration example extracted from [Moreno-Noguer and Fua, 2013]. It shows how different 3D shapes could yield a similar 2D projection, although the first out of the three proposed solution rows is the one with the least 3D error.

### 2.2.1.1 Physics-based priors

In the literature, the following priors have been proposed, based on how real objects deform when forces are applied:

- **Piece-wise planarity:** It assumes the surface deforms locally as a planar surface and it is modeled by a combination of little planar patches [Varol et al., 2009].
- **Partial rigidity:** It assumes that there is a rigid prevalent component on the object and then the rest of non-rigid components are much less significant [Aanæs and Kahl, 2002].
- **Isometry:** This prior implies that the geodesic distance between surface points does not change [Bartoli and Collins, 2013, Chhatkuli et al., 2014a]. It is a strong prior although much weaker than rigidity or local rigidity. In fact, rigidity is a particular case of isometry.

For two neighboring points  $i$  and  $j$  on the surface  $k$ ,  $Q_i^k$  and  $Q_j^k$  are isometric with respect to the points on surface  $l$   $Q_i^l$  and  $Q_j^l$  if:

$$\|Q_i^k - Q_j^k\|^2 = \|Q_i^l - Q_j^l\|^2 \quad (2.8)$$

- **Inextensibility:** it is a relaxation of the isometry prior as it imposes that the euclidean distance between surface points must be less or equal than the geodesic

distance of the points in the original surface [Chhatkuli et al., 2016, Vicente and Agapito, 2012].

If the equality is reached, the isometry is met. This prior assumes  $d_{ij}$  is the geodesic distance between points  $Q_i^k$  and  $Q_j^k$  of surface  $k$ , so it is inextensible if the following condition is met:

$$\|Q_i^k - Q_j^k\|^2 \leq d_{ij} \quad (2.9)$$

This prior is combined with a maximization of point depths to obtain a convex relaxation of isometry, commonly known as the Maximum Depth Heuristic (MDH).

- **Elasticity:** this prior allows the object to undergo elastic deformations. It is weaker than isometry and usually depends on physical parameters of the object, such as the Young’s modulus [Haouchine et al., 2014]. This constraint is usually discretized using Finite Element Method (FEM) methods [Agudo et al., 2012b, Haouchine et al., 2014].

#### 2.2.1.2 Statistics-based priors

The following statistical priors were mainly studied in the literature of deformable reconstruction:

- **Temporal smoothness:** It can be applied to both the camera and the object. The camera is assumed to move smoothly along the sequence and the object deformations are assumed to vary slowly [Torresani et al., 2008].
- **Shape smoothness:** It only applies to the reconstructed object. It assumes that the shape is smooth, so it has no sharp endings, peaks, salients, etc. It also means that the local curvature of the surface is small [Torresani et al., 2008].
- **Low-rank shape:** It is based on the fact that the number of the detected points and the number of frames is much greater than the maximum number of deformations presented on an object. This is the statistical assumption behind the proposal of [Bregler et al., 1999], that will be explained later in 2.2.3.4 section due to its importance in most of the non-rigid reconstruction algorithms and this Thesis in particular.

A more extensive study about these priors can be found in [Paladini, 2011]. As indicated in this work, the priors are applied on both NRSfM and Shape from Template (SfT) problems.

It must be highlighted that the deformation prior depends on the type of object to be reconstructed. For instance, isometry or inextensibility prior could not be applied to



an elastic surface. A temporal smoothing prior could not be applied to a rapidly moving camera along a sequence or to a strongly waving flag, in which the variation changes on the surfaces are substantial. This makes deformable reconstruction a more object specific problem than SfM.

In [Moreno-Noguer and Fua, 2013] there are two clues that are used to disambiguate possible solutions out of the set of possible non-rigid solutions: 1) The shading information for estimating the sources of light from photogrammetric clues, and 2) the temporal consistency of the solutions. As indicated in the paper these clues are used as a source of disambiguation, that could be changed depending on the type on the problem considered.

### 2.2.2 Model-Based reconstruction techniques

The techniques grouped in this section are considered model-based reconstructions or SfT, as they perform 3D reconstruction on a deformable object whose reference model or template is previously known.

The reference model can be computed from samples, from a prior model of a physical-based object, synthetically computed or even generated by other reconstruction algorithms from a training sequence.

The solutions can be grouped in function of the type of deformation priors used.

#### 2.2.2.1 Statistics-based SfT

The most commonly known algorithm on the literature that belongs to this category is the Active Appearance Models (AAM), which is widely used for the reconstruction and tracking of the human face. It was first introduced by [Cootes et al., 1998]. These methods are also known as Morphable models.

The 3D Morphable Models (3DMM) can be seen as a sophisticated variant of the AAM. The main difference is that they are dense, instead of sparse, and try to evaluate the whole 3D shape of the object. An example of a system using Morphable models for dense face tracking is depicted in [Muñoz et al., 2009]. A revisiting example of these models including an extensive learning and its application over a database of 10,000 faces is studied in [Booth et al., 2016].

Another variant of 3DMM is the one presented in [Bernard et al., 2016], this performs a thorough analysis of the shape fitting by Principal Component Analysis (PCA) given by other standard methods, but this algorithm proposes iterative improvement on local support by applying sparsity to the factorization and further applying regularization to the model.

The main disadvantage of AAM and related algorithms resides on the need of large labeled databases for a proper training of the model. Hand-made labeled databases are

prone to errors, not only due to the “human factor”, but also as different persons could do the task differently. In the work of [Zhou et al., 2016] it takes advantage of some of the approaches and dense correspondences to both infer the point correspondences and predict the labeling.

An attempt to introduce Gaussian mixtures on the shape model and then a Kalman filter on the visible regions to stabilize the estimation is done by [Sanchez-Riera et al., 2010].

### 2.2.2.2 Physics-based SfT

Most of SfT methods fall into this category and in particular using the isometry prior [Brunet et al., 2010, Bartoli et al., 2015, Vicente and Agapito, 2012]. [Bartoli et al., 2015] describes the problem as a Partial Differential Equation (PDE) system and proves that imposing the isometry prior makes SfT a well-posed problem. In Fig. 2.8 an illustration of the SfT approach is shown when isometric constraints are imposed.

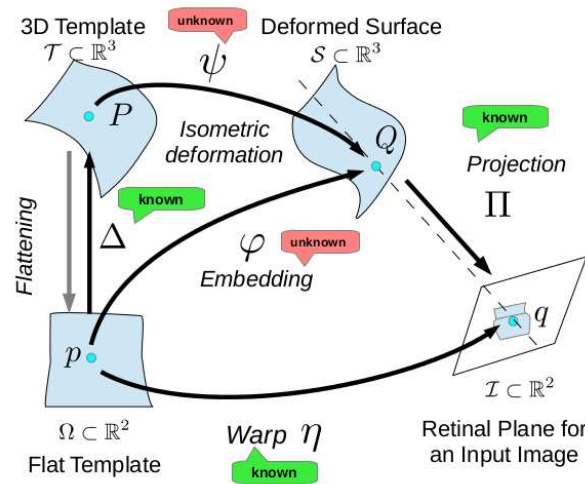


Figure 2.8: Geometric modelling scheme of SfT imposing isometric constraints. Extracted from [Chhatkuli et al., 2014b]

Existing methods in isometric SfT can be divided into local solutions, mainly based on solutions of a PDE system and global solutions, based on convex relaxations of isometry [Fua and Salzmann, 2011, Ngo et al., 2016]. Recently [Gallardo et al., 2016] proposed a dense SfT approach able to capture high-frequency deformations by incorporating shading information. Dense Isometric SfT has been implemented in real-time in [Collins and Bartoli, 2015] using a high-end GPU.

Non-isometric priors, such as elasticity has been recently studied [Haouchine et al., 2014]. Well-posedness is not guaranteed in these methods and they usually require bound-



ary conditions. Topology changes in elastic materials, such as cuts and tearing is modeled in [Paulus et al., 2015].

Few approaches solve SfT from RGBD data. In [Leizea et al., 2014], a similar approach in which tracking and mapping are separated is presented. This makes use of Mass Spring Model (MSM) instead of FEM to make the computations a bit lighter. A Computer Aided Design (CAD) 3D shape is loaded and then its pose and shape is tracked frame by frame, adapting to the deformations.

The main drawback of all the template based approaches is that they require a very strong knowledge about the object to be reconstructed.

In order to better organize the works of the literature, the Table 2.2 is depicted.

Model-based approach	sparse/dense	GPU	proj	seq.	RT	priors	highlights
[Cootes et al., 1998] AAM	sparse	no	orth	yes	yes	N/A	prior training
[Muñoz et al., 2009] 3DMM	dense	no	pers	yes	no	N/A	full tracking of the 3D model
[Booth et al., 2016] 3DMM	dense	N/A	pers	no	no	N/A	
[Bernard et al., 2016] 3DMM	dense	N/A	N/A	no	no	N/A	linear shape deformation models
[Zhou et al., 2016] AAM	sparse/dense	N/A	N/A	no	no	low K	constructs deformable models with shape flow
[Brunet et al., 2010] FFD	sparse	no	pers	no	no	iso	
[Joseph Tan et al., 2014] FFD	sparse	no	N/A	yes	yes	N/A	requires training of the linear predictors
[Salzmann et al., 2008]	sparse	no	pers	no	no	inext	inext meshes, closed form
[Sanchez-Riera et al., 2010]	sparse	no	pers	no	no	pose, shape	iterative solution, priors modeled as gaussian mixtures
[Fua and Salzmann, 2011]	sparse	no	pers	no	no	shape	linear local models
[Pizarro and Bartoli, 2012]	sparse	no	orth	no	no	shape	warp estimation, self occlusion reasoning
[Bartoli and Collins, 2013]	sparse	no	pers	no	no	iso	solution for uncalibrated weak and full perspective
[Chhatkuli et al., 2014b]	sparse	no	pers	no	no	iso	PDE solution for SFT problem
[Bronte et al., 2014] (factorization)	sparse	no	pers	yes	yes	N/A	PTAM + linear bases
[Collins and Bartoli, 2015]	sparse/dense	yes (render)	pers	no	<21fps	iso,thin shell	SfT impt for AR
[Leizea et al., 2015]	sparse	no	N/A	yes	yes	pre-trained deforms	particle filter based tracking for deforms
[Ngo et al., 2016]	sparse	N/A	pers	no	10 fps	reg mat	reg. matrix, planar/curved templ, c++ impl.
[Magnenat et al., 2015]	sparse	no	N/A	yes	27fps	reg mat, time, shape	Live texturing of a drawing book
[Wang et al., 2016]	sparse	no	orth	no	no	N/A	non-rigid point registration
[Leizea et al., 2014]	dense	no	N/A	yes	7-15 fps	MSM	solid tracking based on model and physical model
[Ngo et al., 2015]	dense	no	pers	no	no	length,smooth	performs whole image alignment, minimal texture
[Parashar et al., 2015]	dense	no	pers	no	no	As Rigid As Possible (ARAP)	extension of SfT to volumetric modelling
[Paulus et al., 2015]	sparse/dense	no	N/A	yes	no	stretching,FEM	cutting and tearing object modelling
[Yu et al., 2015]	dense	yes	pers	yes	no	spatial, temp, ARAP	dense tracking from model
[Liu-Yin et al., 2016]	dense	yes	pers	yes	no	spatial,ARAP, temp,sparse, shading,specular	SfT + SfShading
[Gallardo et al., 2016]	dense	N/A	pers	N/A	no	shading,motion, boundary,iso	SfT + SfShading

Table 2.2: Model-based / SfT approaches summary

### 2.2.3 Non-Rigid Structure-from-Motion (NRSfM)

In NRSfM the objective is to recover the 3D shape of an object undergoing deformations from a sequence of images. Each image shows the combination of rigid motion and shape change in the object.

#### 2.2.3.1 Batch NRSfM

Most of batch NRSfM methods use statistics-based priors and in particular the low-rank shape prior [Del Bue et al., 2006, Torresani et al., 2008, Garg et al., 2013, Dai et al., 2012] and the non-rigid factorization algorithm. It was first introduced by [Bregler et al., 1999], modelling deformations as a linear combination of basis shapes, resulting in specific metric constraints on the factorization problem. Several works improved this approach to better constrain the reconstruction. Fig. 2.9 shows the general idea behind low-rank shape models, where the object's shape space is assumed to be low-dimensional and the shape is represented as the weighted sum of a set of basic shapes or shape basis. Most of these methods are based on the orthographic camera.

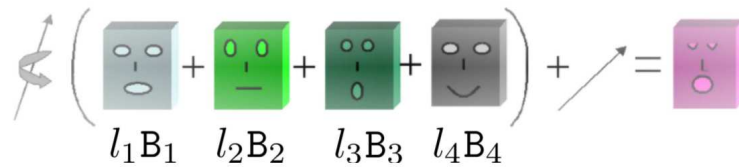


Figure 2.9: Linear basis shapes approximation. A visual representation of the model based on a set of linear bases multiplied by its weights, conveniently rotated and translated in space could yield the image of the current shape. Extracted from [Paladini, 2011].

Non-rigid factorization with the perspective camera was studied in [Lladó et al., 2005] and [Hartley and Vidal, 2008].

It's generally accepted that the low-rank prior renders NRSfM an ill-posed problem. Therefore, the efforts were focused on adding priors on the factorization of the tracking matrix. Some of them were estimating an initial rigid component [Del Bue et al., 2006], setting priors on temporal smoothness or spacial smoothness [Torresani et al., 2008] or using point trajectory constraints [Akhter et al., 2009, Gotardo and Martinez, 2011]. Recently [Dai et al., 2012] demonstrates that many of the ambiguities associated to the low-rank prior can be solved by minimizing the nuclear norm of the shape matrix. Many recent methods use this prior, combined with stronger priors, such as the as-rigid-as-possible soft constraint.

Low-rank shape priors are accurate for objects with simple deformations, such as the human face. Deformations of some objects are high-dimensional. However less-constraining the rank leads to over-fitting and wrong reconstructions [Paladini et al.,

2009]. Physics-based priors, such as isometry, are useful for special deformation cases [Vicente and Agapito, 2012, Brunet et al., 2010].

Other approaches, like [Fayad et al., 2010, Russell et al., 2011] tackled the problem in a piece-wise sense, using local models that better adjust certain parts of an object. The main drawbacks of these approaches are how to assign the initial partition set of models and how to assign the overlapping between points that share different models. This implies that global coherence of the model, even it could be better adjusted, is not guaranteed.

As commented in the rigid section introduction (2.1), there were several approaches for the rigid case that got successful results on monocular dense SfM in real time, as demonstrated by DTAM [Newcombe et al., 2011b] and further extended to KinectFusion [Newcombe et al., 2011a]. Since then, few approaches appeared to tackle the dense case of NRSfM. The pioneering work in this line was the presented in [Garg et al., 2013], which did per pixel dense reconstruction combining low rank prior with local smoothness priors. In [Russell et al., 2014] authors proposed segmentation and reconstruction of local rigid models.

Physics-based NRSfM has been recently studied for the isometric prior [Chhatkuli et al., 2014a, Parashar et al., 2016, Chhatkuli et al., 2016]. [Parashar et al., 2016] proved that isometric NRSfM is a well-posed problem by studying the differential properties of the problem. It also proposes very accurate local solutions. [Chhatkuli et al., 2016] proposes a global reconstruction method using inextensibility and the maximum depth heuristic. It shows that NRSfM admits a convex relaxation that results in a large-scale Second Order Cone Programming (SOCP) problem.

### 2.2.3.2 Sequential NRSfM

Previous approaches are batch and not suitable to run in real time. Very few methods address sequential NRSfM. [Paladini et al., 2010] proposes a low-rank shape prior, that is imposed on a temporal sliding window. [Agudo et al., 2012a, Agudo et al., 2012b, Agudo et al., 2016a] proposes a sequential approach based on elasticity constraints and FEM models (see Fig. 2.10). This is an object specific model that depends on knowledge about physical parameters such as the Young’s modulus. It cannot be considered a pure NRSfM method as it requires a reference shape of the object obtained from rigid SfM.

### 2.2.3.3 Camera models in NRSfM

Nowadays, very few of the approaches tackle the case of the perspective projection, as the projection equations are more complex. Just two factorization based works tackled the

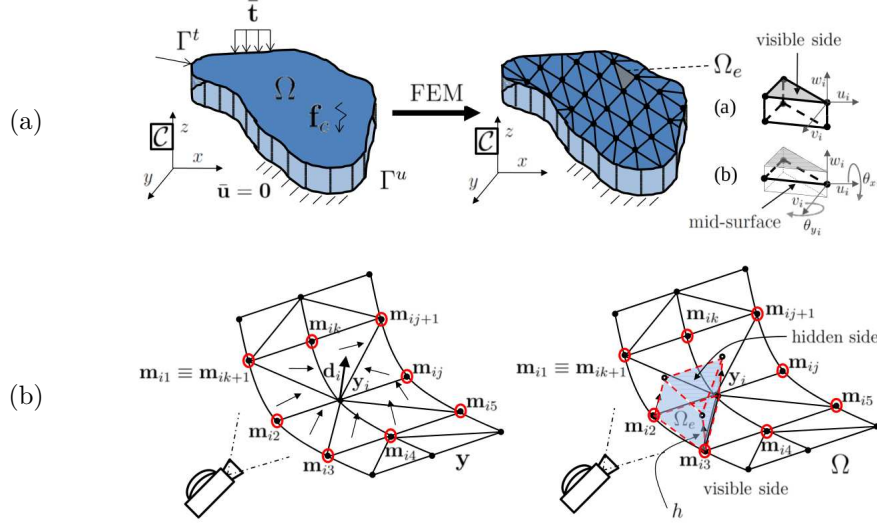


Figure 2.10: FEM modelling principles. (a) shows the FEM partition of a surface or volume into wedges. (b) shows the estimation of the surface normals to estimate volumetric wedges. Both figures are extracted from [Agudo et al., 2016a]

perspective projection [Lladó et al., 2005, Hartley and Vidal, 2008]. Using the assumption that the camera is far enough from the objects to be modeled and that the objects are shallow, the perspective effects can be avoided. In real situations this assumption cannot be taken. In fact in some of the latest works, like [Agudo et al., 2016b, Bartoli and Collins, 2013, Chhatkuli et al., 2014b] use perspective projection.

With this projection model, a more realistic way of modelling the objects can be obtained when the object is close to the camera, as the perspective effects are more evident.

The perspective effects can be used to minimize the effects of the ambiguities previously addressed, except for the scale.

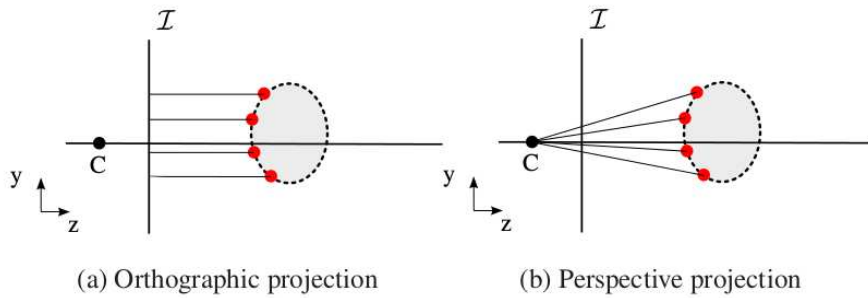


Figure 2.11: Orthographic vs perspective camera projection. Taken from [Paladini, 2011]

It is illustrated on Fig. 2.11 the differences between the orthographic and perspective projections. All the points are projected the same way wherever the projection plane is located in (a), whereas the projections are scaled if the image plane is moved for (b).

A taxonomy of the previous approaches is shown in Table 2.3.

NRSfM approach	sparse/dense	GPU	proj	sequential	real-time	priors	features
[Bregler et al., 1999] (factor)	sparse	no	orth	no	no	no	first factor based work
[Aanas and Kahl, 2002] (factor)	sparse	no	orth	no	no	shape	first work depicting ambiguities
[Torresani et al., 2003] (factor)	sparse	no	orth	no	no	gaussian shape, temp	Expectation-Maximization (EM) estimation scheme
[Lladó et al., 2005] (factor)	sparse	no	pers	no	no	no	divides problem in subproblems, iterative
[Del Bue et al., 2006] (factor)	sparse	no	pers	no	no	rigidity	estimates rigidity for motion independently
[Hartley and Vidal, 2008] (factor)	sparse	no	pers	no	no	no	closed form
[Torresani et al., 2008] (factor)	sparse	no	w-pers	no	no	shape	linear subspace, PPCA
[Paladini et al., 2009] (factor)	sparse	no	orth	no	no	N/A	metric projection, deform and articulated
[Paladini et al., 2010] (factor)	sparse	no	orth	yes	no	temp(rot),shape	first sequential NRSfM approach
[Fayad et al., 2010] (patch)	sparse	no	orth	no	no	temp (pose)	piecewise patch NRSfM
[Russell et al., 2011] (patch)	sparse	no	orth	no	no	temp (pose)	same as before, dynamic model assignment
[Moreno-Noguer and Porta, 2011] (fact)	sparse	no	pers	no	no	statistical	MAP formulation, probabilistic setup
[Dai et al., 2012] (fact,min K)	sparse	no	orth	no	no	min-K,orto	best CVPR12 paper, orth condition inclusion
[Akhter et al., 2009] (traj)	sparse	no	orth	no	no	no	trajectory bases
[Gotardo and Martinez, 2011] (traj)	sparse	no	orth	no	no	time smooth	trajectory bases, prev recon
[Vicente and Agapito, 2012] (fact)	sparse	no	orth/pers	no	no	iso,shape,temp	template based / template free
[Agudo et al., 2012a] [Agudo et al., 2016a] (FEM)	sparse	no	pers	yes	yes(30 pts)	FEM	FEM+EKF-SLAM
[Lee et al., 2013] (procrust)	sparse	no	orth	no	no	procrustean normal	procrustean normal distribution for NRSfM
[Agudo and Moreno-Noguer, 2015a] (fact)	sparse	no	orth	no	no	low rank traj	models force space from deformations
[Agudo and Moreno-Noguer, 2015b] (BA+physics)	sparse	no	orth	yes	no	motion laws, temp(pose), shape, extensibility	movement laws based BA for NRSfM
[Chhatkuli et al., 2016] [Chhatkuli et al., 2014a] (infinitesimal)	sparse	no	pers	no	N/A	iso	inextensible reconstruction SOCP, intrinsic template estimation
[Parashar et al., 2016] (infinitesimal)	sparse	no	pers	no	N/A	iso	infinitesimal planarity based reconstruction
[Kong and Lucey, 2016] (priorless)	sparse	no	orth	no	no	priorless compressible	priorless compressible based on block sparse dictionary learning
[Garg et al., 2013]	dense	yes	orth	no	no	spacial (edge and trace)	first variational NRSfM approach
[Russell et al., 2014]	dense	yes	orth	no	no	edge, sparse (overlap), rigidity, saliency	piecewise, dynamic scene dense reconstruction
[Agudo et al., 2014]	dense/sparse	no	orth	yes	no	FEM	tackling dense data decimating, triangulating and applying FEM
[Newcombe et al., 2015]	dense	yes	pers	yes	yes	N/A	RGBD input, dynamic non rigid model and wrap computation
[Lee et al., 2016]	sparse/dense	no	orth	no	no	no	weak recons, solves reflexion ambiguities and then "strong" recons
[Ranftl et al., 2016]	dense	yes	pers	no	no	no	comparative for dynamic scenes between methods, including NRSfM

Table 2.3: NRSfM approaches summary

#### 2.2.3.4 Bregler et al.’s batch approach

The approach presented in [Bregler et al., 1999] is considered the pioneer method for NRSfM. Most of the works that appeared after, including this Thesis, consider the deformation prior based on low-rank shape basis. Given its importance, its foundations are detailed here.

The main idea is to extend the low-rank factorization performed by [Tomasi and Kanade, 1992], in which the input of the algorithm is the set of 2D aligned tracked points. The correspondence between images, as in most of the related works, is supposed to be given and perfect.

The second assumption in the paper is that the projection model is orthographic. The

tracking matrix  $W$  is defined as follows:

$$W = \begin{pmatrix} q_{11} & \cdots & q_{1N} \\ \vdots & \ddots & \vdots \\ q_{F1} & \cdots & q_{FN} \end{pmatrix} = \begin{pmatrix} l_{11}R_1 & \cdots & l_{1K}R_1 \\ \vdots & \ddots & \vdots \\ l_{F1}R_F & \cdots & l_{FK}R_F \end{pmatrix} \begin{pmatrix} B_1 \\ \vdots \\ B_K \end{pmatrix} = MS, \quad (2.10)$$

where  $q_{fn}$  is  $2 \times 1$  vector that denotes the image projection of the  $n$ th point in the  $f$ th frame. All coordinates of frame  $f$  are centered with respect to their centroid.  $N$  denotes the number of shape points and  $F$  the number of frames. Matrix  $W$  is thus of size  $2F \times N$ . The low-rank shape bases assumes that  $W$  can be decomposed into the multiplication of the  $2F \times 3K$  motion matrix  $M$  and the  $3K \times N$  shape matrix  $S$ , where  $K$  is the number of shape bases considered. The matrices  $\{B_1 \cdots B_K\}$  are the set of  $K$  3D bases shapes, each of them of size  $3 \times N$ .  $l_{fk}$  are the deformation coefficients and  $R_f$  defines the projection matrices of size  $2 \times 3$  (in orthographic projection these are Stiefel matrices, containing the first 2 rows of the rotation matrix, being also orthonormal vectors).

Therefore, for each frame  $f$ , the shape  $S_f$  can be expressed as:

$$S_f = l_{f1}B_1 + \cdots + l_{fK}B_K \quad (2.11)$$

Each 3 consecutive rows of  $M$  are defined as follows:

$$a_f = (l_{f1}R_f \cdots l_{fK}R_f) = \begin{pmatrix} l_{f1}r_{f1} & l_{f1}r_{f2} & l_{f1}r_{f3} & \cdots & l_{fK}r_{f1} & l_{fK}r_{f2} & l_{fK}r_{f3} \\ l_{f1}r_{f4} & l_{f1}r_{f5} & l_{f1}r_{f6} & \cdots & l_{fK}r_{f4} & l_{fK}r_{f5} & l_{fK}r_{f6} \end{pmatrix} \quad (2.12)$$

where

$$a_f = \begin{pmatrix} l_1 \\ \vdots \\ l_K \end{pmatrix} \begin{pmatrix} r_{f1} & r_{f2} & r_{f3} & r_{f4} & r_{f5} & r_{f6} \end{pmatrix} \quad (2.13)$$

being  $r_{fi}$  with  $i = 1, \dots, 6$  the coefficients of the  $f$ th projection matrix  $R_f$ .

Note that the decomposition of  $W$  depicted in equation 2.10 is ambiguous. Given any invertible  $3K \times 3K$  matrix  $G$  and  $W = MS$ , the decomposition  $\hat{M} = MG$  and  $\hat{S} = G^{-1}S$  is also valid. Some of the ambiguities can be solved by forcing  $R_f$  with  $f = 1, \dots, F$  to have orthonormal rows:

$$\begin{cases} \begin{pmatrix} r_{f1} & r_{f2} & r_{f3} \end{pmatrix} G_f G_f^T \begin{pmatrix} r_{f1} & r_{f2} & r_{f3} \end{pmatrix}^T = 1 \\ \begin{pmatrix} r_{f4} & r_{f5} & r_{f6} \end{pmatrix} G_f G_f^T \begin{pmatrix} r_{f4} & r_{f5} & r_{f6} \end{pmatrix}^T = 1 \\ \begin{pmatrix} r_{f1} & r_{f2} & r_{f3} \end{pmatrix} G_f G_f^T \begin{pmatrix} r_{f4} & r_{f5} & r_{f6} \end{pmatrix}^T = 0 \end{cases} \quad (2.14)$$

where  $G_f$  with  $f = 1, \dots, F$  depend on the coefficients of the unknown matrix  $G$ . Solving for  $G$  (metric update) is one of the main problems with this approach. Several methods have been proposed to find  $G$  imposing additional constraints.

### 2.2.3.5 Paladini et al.'s Sequential approach

Based on the previous factorization approach, the idea of a sequential reconstruction of deformable objects, inspired by PTAM [Klein and Murray, 2007] and imitating the way in which rigid sequential SLAM systems were acting was proposed in [Paladini et al., 2010]. Before this method all the available approaches were batch, so it was the first proposal able of doing an incremental reconstruction.

This algorithm proposed a rank-growing engine to check and insert when it was needed a new deformation mode, inside a sliding window scheme. It also proposed a model capable of representing deformations of rank lower than 3, for instance, along a plane or a line. Therefore, this approach is able to deal with degenerate deformations.

The NRSfM algorithm proposed in [Paladini et al., 2010] obtains first a rigid shape estimation using a factorization over the first few frames of the sequence. Then, the process consists of using a non-rigid BA over a sliding window of  $w$  frames.

The cost to be minimized was the following:

$$\min_{R_i, U_i} \sum_{i=f-W}^f \|W_i - R_i (S + U_i V)\|_F^2 \quad (2.15)$$

where  $R_i$  is the orthogonal projection 2x3 matrix,  $W_i$  is a selection of the latest frames of the tracking matrix, and  $\|\cdot\|_F$  is the Frobenius norm.

The method proposed optional priors, such as rotation and shape smoothness. Therefore, the expression to be minimized including these priors was as follows:

$$\min_{R_i, U_i} \sum_{i=f-W}^f \|W_i - R_i (S + U_i V)\|_F^2 + \lambda \sum_{i=f-W}^f \|R_i - R_{i-1}\|_F^2 + \psi \sum_{i=f-W}^f D_{i,i-1} \quad (2.16)$$

If the error could not be sufficiently minimized using this expression, the model needed to be updated. From the current estimate of  $U_{f,1:r-1}$  and  $V_{1:r-1}$  the error of the model was computed as

$$\widetilde{W}_f = R_f \left( \bar{S} + U_{f,1:r-1} V_{1:r-1} + U_{f,r} V_r \right) \quad (2.17)$$

$$A = \widetilde{W}_f - R_f \left( \bar{S} + U_{f,1:r-1} V_{1:r-1} \right) \quad (2.18)$$

where  $Z = U_{f,r} V_r$ .  $A$  was needed such as  $A = R_f Z$  and  $\text{rank}(Z) = 1$ . As it was difficult to be solved, a linear solution based on linear least squares was taken and then the first row and column is taken from a Singular Value Decomposition (SVD) decomposition to constrain the rank to 1.

After the rank growth, the sliding window was run again with the new set of bases.

Even though it was not said in the paper, there was an important issue to remark in the implementation. When trying to reduce the error it could happen that the amount of bases to describe a deformation of an object could grow indefinitely. In order to sort this problem out, a re-factorization is proposed when a certain rank is reached.

The re-factorization of the bases consists of computing PCA over the current existing ones and taking only the most important ones, discarding the rest and re-computing the coefficients of the rest of the reconstructions.

Although this work is seminar in terms of being the first sequential NRSfM approach and having served as inspiration to many other works, including the present Thesis, the idea of making the estimations within a sliding window has been tackled by other authors [Agudo et al., 2016a, Agudo et al., 2014, Agudo et al., 2012a] and their results outperformed the original approach for certain type of objects.

In order to implement re-factorization proposal, the shape was reorganized, instead of factorizing a  $S_{F \times 3P}$  matrix, the shape matrix was set to  $S_{3F \times P}$ , being  $F$  the number of frames and  $P$  the number of points, where each of the bases were transformed from  $3 \times P$  to  $1 \times P$  each, so the coefficients were differentiated by each spacial dimension. The mean shape was also considered in the deformation model, as usually done, so the model was given by:

$$S_f = \bar{S} + \begin{pmatrix} U_{f1} & \cdots & U_{fr} \end{pmatrix} \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_r \end{pmatrix} \quad (2.19)$$

in which  $\bar{S}$  was the mean shape, the  $U_{fr}$  were the 3-vector  $U_{fr} = \begin{pmatrix} U(x)_{fr} & U(y)_{fr} & U(z)_{fr} \end{pmatrix}^T$  and  $V_r$  the rows of the  $V$  matrix. It can be considered that  $V$  was the low rank implicit model after a PCA compression and the  $U$  matrix was the 3 dimensional coefficients affecting the bases to reconstruct the shape, represented in  $3F \times P$ .



Therefore, the tracking matrix given in orthographic projection, could be given by the following expression

$$W_f = \begin{pmatrix} u_{f1} & \cdots & u_{fP} \\ v_{f1} & \cdots & v_{fP} \end{pmatrix} = R_f S_f + T_f = R_f (\bar{S} + U_f V) + T_f \quad (2.20)$$

Using a rank 1 approximation to the model instead of the standard rank 3, the algorithm becomes ideal to model degenerate deformations (1D or 2D ones), although this part is not the most interesting one as the most frequent deformations that appear on standard sequences are rank 3, which would imply growing three times the rank of the model per frame, which is, in terms of CPU consumption very expensive.

Some results for the described algorithm are shown in Fig. 2.12 both quantitative and qualitative for ideal tracking condition even though missing data is considered on the tracking.

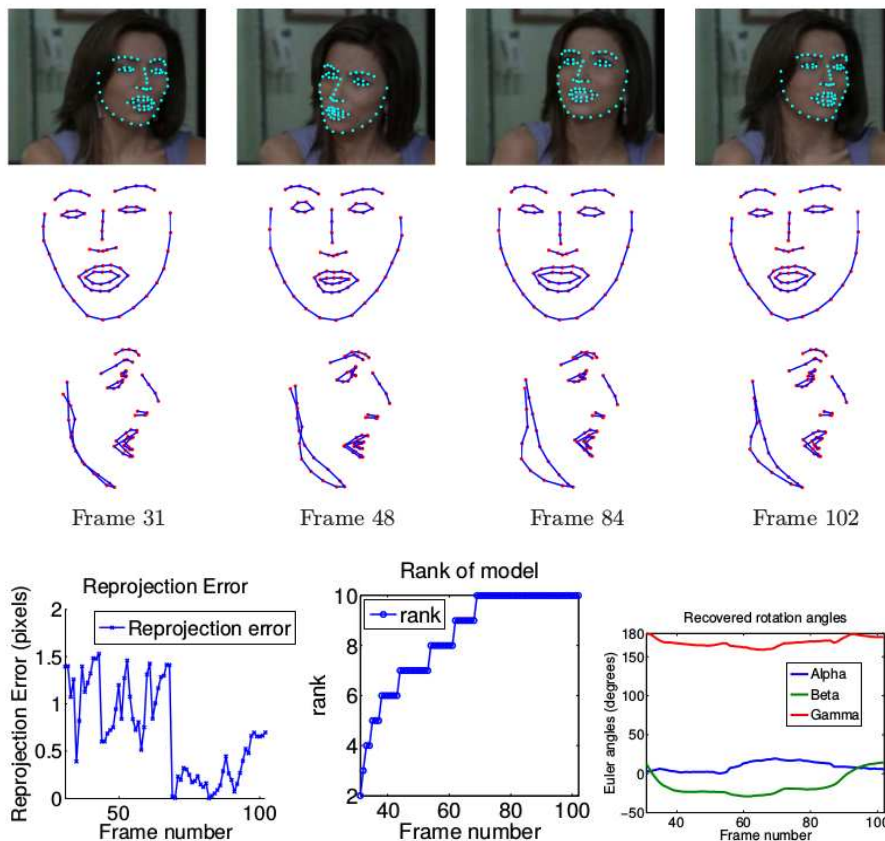


Figure 2.12: Sequential NRSfM results on actress sequence. On top qualitative results and at the bottom, quantitative results, including *reprojection* error, rank growth and recovered rotation angles.

As the approach is orthographic no information about translation is given.

## 2.3 Thesis objectives

This thesis takes the philosophy of two well known methods from the state-of-the-art, one rigid and one non rigid, and combine both to propose a sequential NRSfM approach.

The first is PTAM [Klein and Murray, 2007], an efficient method in terms of processing time, which provides a rigid framework to modify and deploy new algorithms in c++. PTAM is efficient in terms of processing time. This provides a good reference for the state-of-the-art, as it has been well tested, it has been already compared with other works [Davison et al., 2007, Newcombe et al., 2011b, Newcombe et al., 2011a], and it has also been used as base of other works [Pan et al., 2009, Newcombe and Davison, 2010, Stühmer et al., 2010, Tan et al., 2013].

The second is the sequential and incremental non-rigid processing framework of [Paladini et al., 2010]. The main idea behind the algorithm is to learn how an object is deforming in a incremental and sequential way. The algorithm uses a two step error optimization scheme in which, for each frame, the tracking process finishes and sequentially the modelling update starts.

Hereafter the concrete objectives to be tackled in this thesis are enumerated:

### 2.3.1 Non rigid tracking

This objective is to modify the core of PTAM tracking system, to adapt it to handle a non-rigid model, maintaining real-time performance and its main features.

### 2.3.2 Incremental Modeling

Following the same philosophy of PTAM we propose a mapping process that runs in parallel with the tracking. In this case the mapping consists of a batch NRSfM algorithm that can perform a fast and good reconstruction for a window of keyframes in a sequential way, in the line of the work of [Paladini et al., 2010].

### 2.3.3 Parallel tracking and modelling on collaborative mode.

Both threads will be adapted to work with non-rigid sequences, the update between both threads will be transparent. The tracking thread keeps the model on real time and checks if the currently seen deformations can be explained by the low-rank model. If not, new keyframes are sent to the map and the model is conveniently updated. The mapping thread checks for updates and performs batch updates on the incoming data so as to keep a good enough model.

### 2.3.4 Perspective projection

The direct use of perspective projection in the algorithm will allow to be more accurate, reducing the natural ambiguities of the orthographic projection that exist in most of the state-of-the-art NRSfM algorithms.

### 2.3.5 Dealing with real data association

The rejection of missing data and outliers is tackled in very few methods of the state-of-the-art. However, it is a crucial problem when facing tracking of features in a real sequence. In this thesis this problem will be faced.



## Chapter 3

# Real-Time Model-Based Non-Rigid Tracking

This chapter presents a real-time solution for sequential model-based 3D reconstruction of deformable objects. Our solution deals with data association and works in real time. This is also an important module in our sequential Non-Rigid Structure from Motion (NRSfM) solutions, that follows the parallel tracking and mapping philosophy, successfully introduced in Parallel Tracking And Mapping (PTAM) [Klein and Murray, 2007] to solve rigid Simultaneous Localization And Mapping (SLAM).

3D reconstruction from images is a key technology in many applications such as Human-Machine Interface (HMI), Augmented Reality (AR) and robotics. These applications require accurate and stable reconstructions. Deformable reconstruction is needed when dealing with non-rigid objects, such as the human body or tissue. Robust and sequential 3D reconstruction is a priority in most of these cases. Algorithms must be able of dealing with real data association between images, which causes outliers and missing data. The development of a real-time non-rigid 3D reconstruction technique from monocular sequences is clearly motivated in this context.

Real time constraints are defined here by the interval between incoming frames from a camera, which, depending on its quality and acquisition features could vary between 17 ms ( $\sim 60$  fps) to 66 ms (15 fps), being the standard value considered in the literature for most of the capture systems 33 ms (30 fps).

As was discussed in Chapter 2, non-rigid reconstruction counts with two main categories of reconstruction methods: Model-free and Model-based approaches.

Model-free approaches solve the NRSfM problem. They have been studied for both batch [Paladini et al., 2009, Moreno-Noguer and Porta, 2011, Gotardo and Martinez, 2011, Torresani et al., 2003] and sequential [Paladini et al., 2010, Agudo et al., 2012a] reconstructions. They do not need a previous model and assume the low-rank shape basis as the deformation constraint. As was mentioned in Chapter 2, those methods

require accurate feature correspondences between frames, not handling properly outliers and missing data in most of the cases. They are also computationally complex. Even using GPU implementations they cannot produce reconstructions for every frame in a long sequence.

Model-based / Shape from Template (SfT) approaches have been thoroughly studied. In particular, physics-based deformation priors have attracted a lot of attention, such as isometry or elastic deformations [Fua and Salzmann, 2011, Bartoli and Collins, 2013, Parashar et al., 2015, Chhatkuli et al., 2014b, Ngo et al., 2016]. Statistical-based methods have been also proposed, with special attention to particular objects, such as the human face [Muñoz et al., 2009, Cootes et al., 1998]. They are able to simultaneously track feature points from images and reconstruct the 3D shape by using a large training set of labeled image data to train the model. Such training images are synthetically created or often manually labeled. All these works have in common the need of a template, which consists of a reference shape of the object and a texturemap. These methods require registration between the input image and the template’s texturemap to obtain the reconstruction. Deformation priors are then used to reconstruct the 3D shape from the template-to-image registration. Model-based approaches are usually affordable to be run in real-time.

This chapter is organized as follows: our algorithm description following the most basic modelling and projection equations are described in section 3.1, going through the data association between the detected points and the model given in section 3.1.1, as well as the feature matching used in 3.1.2, and the camera motion model in 3.1.3. Afterwards the minimization procedure of the reprojection error is addressed in 3.1.4, as well as the prior integration on that scheme in 3.1.5. In case the tracking algorithm is lost, the basic recovery procedure is depicted in 3.1.6. In section 3.2 a summary of the results is described. Finally, section 3.3 presents the main conclusions of the chapter.

### 3.1 Algorithm Description

Given a video sequence, a set of deformation bases, an initial estimation of the pose and the rigid shape of an object, our algorithm estimates the camera pose and the 3D model deformations of the object in the sequence for each frame, tracking some features over the images. We refer this method as the *tracking thread*, as it will play that role inside our parallel tracking and mapping NRSfM solution.

A block diagram of the tracking thread and the initialization process is presented in Fig. 3.1. The main blocks of our tracking thread are included inside the red box. The yellow blocks represent inputs, the orange blocks are configurations (priors), the green blocks are estimations from the input data and finally the blue blocks are implemented tasks.

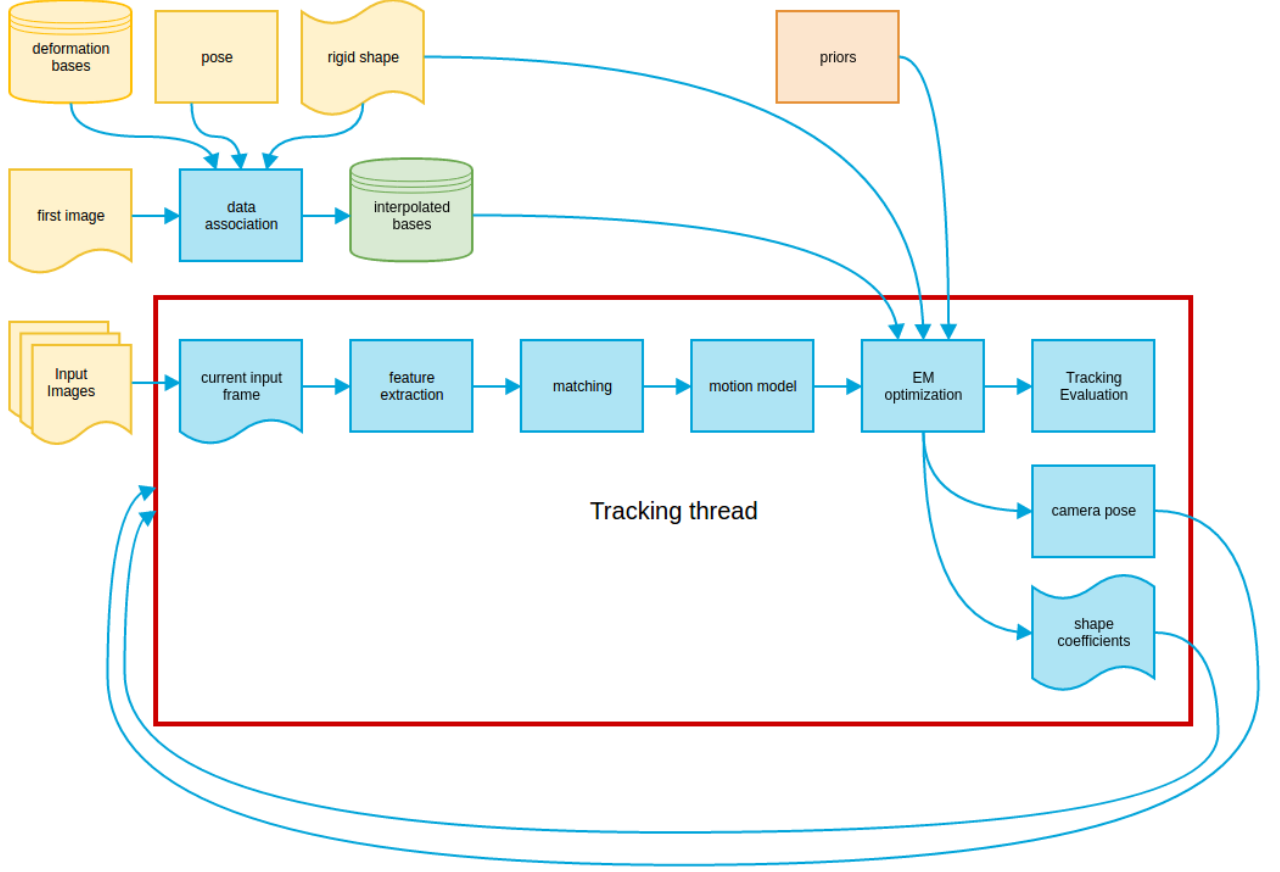


Figure 3.1: Tracking thread block diagram

We use the linear shape basis model introduced by [Bregler et al., 1999] for modelling deformations. The shape is expressed as a linear combination of a fixed set of *basis shapes* multiplied by time-varying coefficients:

$$S_{NR}(f) = S_R + \sum_{k=1}^K L_k(f) B_k \quad (3.1)$$

where  $S_{NR} \in \mathbb{R}^{3 \times P}$  is the (time-varying) shape for frame  $f$ , where  $P$  is the number of points of the considered shape.  $S_R \in \mathbb{R}^{3 \times P}$  is the average rigid shape,  $L(f) \in \mathbb{R}^K$  are the  $K$  deformation weights for frame  $f$ , and  $K$  is the number of basis shapes. The basis shapes  $B \in \mathbb{R}^{3K \times P}$  are fixed for the whole sequence and are assumed to be known, normally learned in an initial set-up from a set of 3D training data.

Each point  $i$  of the non-rigid shape  $S_{NR}$  is transformed from world to camera coordinates as

$$X_i = \begin{pmatrix} x_i & y_i & z_i \end{pmatrix}^T = [R|T] * S_{NR_i} \quad (3.2)$$

The camera extrinsic parameters are represented by the  $4 \times 3$  transformation matrix

$[R|T] \in \text{SE}(3)$ , where  $R \in \mathbb{R}^{3 \times 3}$  specifies camera rotation, and  $T \in \mathbb{R}^3$  is the camera translation vector. Points transformed in the camera coordinates are projected with perspective projection as:

$$U_i^S = \begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \alpha \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \begin{pmatrix} x_i/z_i \\ y_i/z_i \end{pmatrix} \quad (3.3)$$

$U^S \in \mathbb{R}^{2 \times P}$  represents the set of shape points,  $(u_0, v_0)^T$  are the camera center coordinates,  $(f_u, f_v)$  the focal length, known from the prior calibration process, and  $\alpha$  the radial distortion function, described in [Klein and Murray, 2007].

### 3.1.1 Measurement model / data association

The measurement model is based on the detection of sparse features in the image using state-of-the-art feature detectors such as the FAST [Rosten and Drummond, 2006] detector. The most challenging task in the tracking is the association between detected features and projected shape points. In order to address this task, the initial non-rigid shape is used as a template.

The points of the model ( $S_{NR}$ ) are projected onto the image plane, and a Delaunay triangulation is computed using CGAL library [CGAL, 2007] with these projections, resulting in a set of connected vertices  $v_i$ . Feature points will not match the vertices of the model, but can be expressed using barycentric coordinates. A detected point  $U'_i = (u'_i, v'_i)^T$  has barycentric coordinates  $(a_i, b_i, c_i)$  computed as in [Moreno-Noguer and Porta, 2011]:

$$\begin{pmatrix} a_i \\ b_i \\ c_i \end{pmatrix} = \text{pinv} \begin{pmatrix} v_{i,1x} & v_{i,2x} & v_{i,3x} \\ v_{i,1y} & v_{i,2y} & v_{i,3y} \end{pmatrix} * \begin{pmatrix} u'_i \\ v'_i \end{pmatrix} \quad (3.4)$$

Where  $\text{pinv}(\cdot)$  is the matrix pseudo-inverse. In order to better explain the association, it is represented in Fig. 3.2. A triangle of the mesh generated by the triangulation from the model points is represented in the black points, whereas a detected point in the image is represented in red and each of the elements are included in the representation.

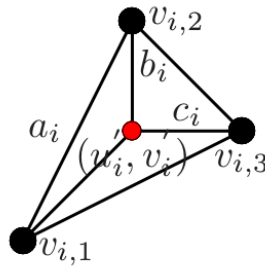


Figure 3.2: Tracking data association from detected points (red) with model points in mesh (black)



Using the barycentric coordinates we can define a set of interpolated basis in world coordinates,  $B'$ , such that the 3D position of a feature point  $X'_i$  relates to the model basis shapes as:

$$X'_i = a_i X_{v1} + b_i X_{v2} + c_i X_{v3} \quad (3.5)$$

$$X'_i = \sum_k L_k (a_i B_{v1k} + b_i B_{v2k} + c_i B_{v3k}) = \sum_k L_k B'_{ik} \quad (3.6)$$

where  $X'_i$  is the interpolated point corresponding to the detected point  $U'_i$ .  $X_{v1}, X_{v2}, X_{v3}$  are the 3D coordinates of the 2D associated coordinates vertices  $(v_{i,1}, v_{i,2}, v_{i,3})$  and  $(B_{v1k}, B_{v2k}, B_{v3k})$  are k-th bases of each of the vertices of the triangle.

Equation 3.6 shows the duality of working with model or detected points and interpolated 3D points, once the data association is set.

### 3.1.2 Feature matching

The quality of the feature matches between the reference image and the rest of the images of the sequence have an impact on the performance results, although a compromise between accuracy and processing time is required not to introduce high delays in the processing pipeline, which could be critical when processing real time sequences.

Two different feature matching methods are studied. The first one is based on PTAM matching and the FAST method and presented in [Bronte et al., 2014]. This is considered as the baseline for the comparison. A second approach based on different state-of-the-art visual descriptors is presented and adapted to work inside the PTAM method.

#### 3.1.2.1 PTAM based matching approach

Feature points detected in the image ( $U'$ ) from FAST [Rosten and Drummond, 2006] correspond to the vertices of  $B'$  (the interpolated bases) because they are easier to track than the points given by the known basis shapes model ( $B$ ). This soft constraint applied to the detected points provides flexibility to the tracking and facilitates the matching of the points among frames, as there is no need to look strictly for features near the initial model points, but we use directly the detected points.

The detected features points on a frame are matched with those detected in the previous frame by using an algorithm similar to the one used in PTAM [Klein and Murray, 2007]. However, the following modifications are needed to deal with non-rigid features:

- First, an affine warping is applied (based on point, patch and pose), like in [Klein and Murray, 2007]. Most of the nearly rigid points will be found using this approach.

- If this matching fails, it is usually due to deformations in the search area. In this case, a multilevel correlation-based approach is performed. The feature in the previous frame is looked up and matched in the top level of the current frame pyramid, and refined by matching it in the lower pyramid levels. Matching is discarded if correlation is too low or if displacement is too large. Married matching (current with respect to previous frame) is applied to discard false positives. More information of this method can be found in [Bronte et al., 2014]

This matching approach could fail for points with high deformed texture. The effect of these points is mitigated by using all available matches of the model estimation in the calculation of the deformation coefficients. This approach does not work for points that move far away from frame to frame (when the searching area is broad), as the matching quality can be low and it can lead to errors in the next estimation stages.

### 3.1.2.2 Descriptor based matching approach

The detection and matching layer is substituted by descriptor based features implemented in OpenCV [Itseez, 2015]. Some of them are directly available, like KAZE [Alcantarilla et al., 2012], AKAZE [Alcantarilla et al., 2013], ORB [Rublee et al., 2011], BRISK [Leutenegger et al., 2011], and others are not directly included but can be accessed as third party software for research purposes, like SIFT [Lowe, 2004] and SURF [Bay et al., 2006]. Binary descriptors are preferred for a real-time implementation as they are faster than the classical ones.

Once the features are detected and described, they must be matched among frames. Several matching algorithms are used such as: *bruteforce*, L1, L2 and Hamming distance. Even though these techniques are reliable enough, some matches could be inexact.

Matches are searched inside a circular area defined in the image domain. The criterion we follow is to select matches of minimum descriptor distance that don't violate the maximum radius condition. The radius parameter is configurable, then, it can be tuned if the number of tracking failures is high.

Feature descriptors, such as SIFT or KAZE are not completely affine invariant and thus can fail with high deformations. Specialized feature descriptors exist for deformable registration, such in [Joseph Tan et al., 2014, Simo-Sierra et al., 2015]. We assume in this thesis that strong deformations are not present, so these options are not considered in this work.

### 3.1.3 Motion model

In order to improve convergence in the tracking, a linear motion model of the camera is computed. The pose is updated using the following motion model:

$$\begin{aligned} vel_t &= \beta/2 (vel_{t-1} + \mu) \\ \widehat{[R|T]}^t &= \exp(vel_t \Delta t) [R|T]^{t-1} \end{aligned} \quad (3.7)$$

where  $vel_t$  is the camera speed at frame  $t$  calculated using the ESM homography between frames [Benhimane and Malis, 2007],  $\beta$  is the factor that modules the influence of  $vel$  on the update, and  $\mu$  is the camera motion vector. The choice of the selected algorithm comes from the original PTAM tracking in [Klein and Murray, 2007]. It assumes a fixed shape, which can be acceptable for our proposal because we don't need an accurate model.

### 3.1.4 E-M optimization

Several estimation methods implement an Expectation-Maximization (EM) approach to compute deformation weights, keeping the camera parameters fixed and vice-versa, like in [Torresani et al., 2003, Torresani et al., 2008, Dellaert, 2002, Agudo et al., 2014].

The Maximum Likelihood Estimation (MLE) function is defined as,

$$f_{MLE}(U', \mu, L, B', S_R) \propto \sum_i |U'_i - proj(X'_i)|^2 \quad (3.8)$$

The parameters we want to estimate are the camera pose vector  $\mu = (\phi_x \phi_y \phi_z t_x t_y t_z)$  composed by the angles and the translations in the 3 axes, and the shape coefficient vector  $L$  for each frame.

We minimize Eq. (3.8) w.r.t. the state vector  $\theta = \begin{bmatrix} \mu & L \end{bmatrix}$ , formed by  $6 + K$  components of pose and deformation weights. We assume  $B'$ ,  $S_R$  and  $K$  fixed.

We run a maximum of 10 EM iterations per frame, alternating between camera pose and deformation coefficients optimization. The current number of iterations depends on the RMS reprojection error. If the error is not significantly reduced between frames the process is stopped. If the error increases, the whole optimization algorithm is applied and, if the error continues increasing, the estimation is stopped. The best solution for the state vector is kept for the next frame. Each estimation is performed by Weighted Least Squares (WLS) minimization of (3.8).

#### 3.1.4.1 E-Step. Deformation estimation

In this step, the goal is to estimate the set of  $K$  deformation coefficients to improve the posterior pose estimation. A preliminary set of coefficients is computed with the current estimation of  $\mu^{t-1}$ ,  $S_R$  and  $B'$ .

The E-step computes the lower bound for the observed data likelihood  $f_{MLE}$  given the previous  $(t - 1)$  state vector. The M-step will find the best camera pose given  $L$ . The

bound is obtained by minimizing the reprojection error over the current set of coefficients  $L$ , and taking into account the motion model:

$$e_i^L = \begin{pmatrix} u'_i \\ v'_i \end{pmatrix} - \text{proj} \left( \widehat{[R|T]}^t \sum_k L_k^{t-1} B'_{k,i} \right) \quad (3.9)$$

Where  $L^{t-1}$  is the current value for the coefficients,  $e_p^L$  the reprojection error for each point  $p$  in the current frame. We weight the observed points using the Tuckey bi-weight M-estimator function [Tukey, 1960] of the reprojection error, with a median-based estimation of the standard deviation, in the same way that the proposed in [Klein and Murray, 2007]. The M-estimator reduces outlier and noise influence on the results, weighting each measurement depending on its reprojection error. Only the detected points from the matching are taken into account.

To compute the deformation weights, starting from Eq. (3.3), each 2D measurement is undistorted, and project the 3D shape on the image plane. For the orthographic case, the 2D camera coordinates are approximated to the first 2 transformed coordinates, and the following expression is valid:

$$\text{proj} \left( \sum_k L_k B'_k \right) = \sum_k L_k \text{proj} (B'_k) \quad (3.10)$$

However, for the case of perspective projection, Eq. (3.10) is not valid because projected points depend also on the depth, as shown in Fig. 2.11 in Chapter 2, which shows a comparative between the two projections. To compute the coefficients, we start from Eq. (3.2) and Eq. (3.3), expanding the projection function we reach:

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u \left( \sum_k L_k \vec{r}_x B'_k + t_x \right) + u_0 \lambda \\ f_v \left( \sum_k L_k \vec{r}_y B'_k + t_y \right) + v_0 \lambda \\ \sum_k L_k \vec{r}_z B'_k + t_z \end{pmatrix} \quad (3.11)$$

where  $R = \begin{pmatrix} \vec{r}_x & \vec{r}_y & \vec{r}_z \end{pmatrix}^T$ ,  $T = \begin{pmatrix} t_x & t_y & t_z \end{pmatrix}^T$  are the camera pose and  $\lambda$  the projection scale.

By grouping the terms  $L_k$  and writing the system in a Linear Least Squares (LLS) form  $Ax = C$ , where  $x = L$ ,  $\Delta u_p = u_p - u_0$ , and  $\Delta v_p = v_p - v_0$ , we obtain:

$$A = \begin{pmatrix} (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{11} & \cdots & (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{1K} \\ (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{11} & \cdots & (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{1K} \\ \cdots & \cdots & \cdots \\ (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{P1} & \cdots & (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{PK} \\ (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{P1} & \cdots & (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{PK} \end{pmatrix} \quad (3.12)$$

$$C = \begin{pmatrix} f_u t_x - t_z \Delta u_1 \\ f_v t_y - t_z \Delta v_1 \\ \vdots \\ f_u t_x - t_z \Delta u_P \\ f_v t_y - t_z \Delta v_P \end{pmatrix} \quad (3.13)$$

Appendix C details how to obtain  $A$  (Eq. 3.12) and  $C$  (Eq. 3.13) from Eqs. (3.1), (3.2) and (3.3).

In the case that an average shape is given for the first frame ( $S_R$ ), this will help to improve the algorithm convergence, describing it as a function of the original basis shapes:

$$S_R = \sum_k L_{R_k} B_k \quad (3.14)$$

where  $L_R \in \mathbb{R}^K$  is the vector of coefficients that describe the average rigid shape. Computing the transformation  $L_{NR} = L - L_R$ , we remove the rigid shape influence.

If the previous condition does not hold, the rigid shape must be introduced in the deduction of the matrices  $A$  and  $C$ , as the 3D model and then the projection equation needs to be adapted.  $A$  matrix remains the same as in Eq. (3.12), but  $C$  gets more complex, as indicated in Eq. (3.15).

$$C = \begin{pmatrix} f_u t_x - t_z \Delta u_1 + S_{R,1} (f_u \vec{r}_x - \vec{r}_z \Delta u_1) \\ f_v t_y - t_z \Delta v_1 + S_{R,1} (f_v \vec{r}_y - \vec{r}_z \Delta v_1) \\ \vdots \\ f_u t_x - t_z \Delta u_P + S_{R,P} (f_u \vec{r}_x - \vec{r}_z \Delta u_P) \\ f_v t_y - t_z \Delta v_P + S_{R,P} (f_v \vec{r}_y - \vec{r}_z \Delta v_P) \end{pmatrix} \quad (3.15)$$

For more details about the whole deduction process we remit the readers to Appendix C.

#### 3.1.4.2 M-Step, Pose estimation

In this step, we compute the camera pose by maximizing the likelihood shown in Eq. (3.8) of the observed data.

The maximum likelihood pose is estimated while keeping the 3D shape fixed, again minimizing the reprojection error:

$$e_i^\mu = \begin{pmatrix} u'_i \\ v'_i \end{pmatrix} - \text{proj} \left( \exp(\mu^t) [\widehat{R|T}]^t \sum_k L_k^t B'_k \right) \quad (3.16)$$

Eq. (3.16) is similar to Eq. (3.9), but in this case we search for the minimum w.r.t.  $\mu$  instead of  $L$ .

We compute the  $\mu$  update in the same way as it's done in [Klein and Murray, 2007, Klein, 2006], as it will be explained below. Similarly to the E-step, the missing correspondences are taken out from the pose estimation on the current frame.

Since the pose parameters are not linear, this time the estimation cannot be done in closed form. Instead it is based on several steps of gradient descent, for which the pose update with respect to the error must be obtained. To that end, Eq. 3.3 is decomposed in several matrices to facilitate the derivation of the expression using the chain rule.

$$\begin{pmatrix} u'_i \\ v'_i \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \end{pmatrix} \begin{pmatrix} A_1 \\ A_2 \\ 1 \end{pmatrix} \quad (3.17)$$

where  $A_i$  are intermediate matrices that will be further defined.

Then the rest of the matrices are concatenated to form the projection expression:

$$\begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = \frac{\tilde{r}}{r} \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \quad (3.18)$$

$$\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} = \begin{pmatrix} \frac{x_c}{z_c} \\ \frac{y_c}{z_c} \end{pmatrix} \quad (3.19)$$

$$r = \sqrt{C_1^2 + C_2^2}, \tilde{r} = r - \beta_1 r^3 - \beta_2 r^5 \quad (3.20)$$

$\beta_1$  and  $\beta_2$  are distortion coefficients previously computed from camera calibration.  $(x_c, y_c, z_c)$  are the 3D points in the camera coordinates.

$$\hat{B} = \frac{\tilde{r}}{r} \Rightarrow \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = \hat{B} \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \quad (3.21)$$

Then, the Jacobians for each of the matrices are defined as follows:

$$J_A = \begin{pmatrix} \frac{\partial u}{\partial A_1} & \frac{\partial u}{\partial A_2} \\ \frac{\partial v}{\partial A_1} & \frac{\partial v}{\partial A_2} \end{pmatrix} = \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \quad (3.22)$$

$$J_{\hat{B}} = \begin{pmatrix} \frac{\partial \hat{B}}{\partial C_1} & \frac{\partial \hat{B}}{\partial C_2} \end{pmatrix} = \begin{pmatrix} -2\beta_1 C_1 - \beta_2 (4C_1^3 + 4C_1 C_2^2) \\ -2\beta_1 C_2 - \beta_2 (4C_2^3 + 4C_2 C_1^2) \end{pmatrix}^T \quad (3.23)$$

$$J_C = \begin{pmatrix} \frac{\partial A_1}{\partial C_1} & \frac{\partial A_1}{\partial C_2} \\ \frac{\partial A_2}{\partial C_1} & \frac{\partial A_2}{\partial C_2} \end{pmatrix} = \begin{pmatrix} \hat{B} & 0 \\ 0 & \hat{B} \end{pmatrix} + \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} J_{\hat{B}} \quad (3.24)$$

$$J_D = \begin{pmatrix} \frac{\partial C_1}{\partial x_c} & \frac{\partial C_1}{\partial y_c} & \frac{\partial C_1}{\partial z_c} \\ \frac{\partial C_2}{\partial x_c} & \frac{\partial C_2}{\partial y_c} & \frac{\partial C_2}{\partial z_c} \end{pmatrix} = \begin{pmatrix} \frac{1}{z_c} & 0 & \frac{-x_c}{z_c^2} \\ 0 & \frac{1}{z_c} & \frac{-y_c}{z_c^2} \end{pmatrix} \quad (3.25)$$

$$\frac{\partial X_c}{\partial \mu_p} = G_p E_{CW} X_W \quad (3.26)$$

with  $p \in 1 \dots 6$  denoting the pose degrees of freedom and  $G_p$  the generation matrix of the  $\mathbb{SE}(3)$  Lie group:

$$\begin{aligned} G_1 &= \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, G_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, G_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\ G_4 &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, G_5 = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, G_6 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{aligned} \quad (3.27)$$

and,  $E_{CW}$  is the pose matrix:

$$E_{CW}|_{4 \times 4} = \begin{pmatrix} R & T \\ 0_{1 \times 3} & 1 \end{pmatrix} \quad (3.28)$$

with  $R$  the  $3 \times 3$  rotation matrix and  $T$  the  $3 \times 1$  translation vector. Therefore, the final expression for the error derivatives with respect to the pose parameters is:

$$\left. \frac{\partial e_i^\mu}{\partial \mu_p} \right|_{2 \times 1} = J_A J_C \begin{pmatrix} J_D & 0 \\ 0 & 0 \end{pmatrix} G_p E_{CW} X_W \quad (3.29)$$

Some of the Jacobians can be grouped and computed in advance:

$$J_{cam}|_{2 \times 2} = J_A|_{2 \times 2} J_C|_{2 \times 2} \quad (3.30)$$

Therefore,

$$\left. \frac{\partial e_i^\mu}{\partial \mu_p} \right|_{2 \times 6} = J_{cam}|_{2 \times 2} \begin{pmatrix} J_{D_i}|_{2 \times 3} & 0 \\ 0 & 0 \end{pmatrix} G_p|_{4 \times 4, p=1..6} E_{CW}|_{4 \times 4} X_W|_{4 \times 1} \quad (3.31)$$

This Jacobian is computed for each point  $X_i'$  in world coordinates ( $X_W$ ) and then inserted into a WLS minimizer together with a M-estimator, for a convenient weight estimator for outlier rejection.

Once the update  $\mu$  for the pose is obtained, as the pose belongs to the  $\mathbb{SE}(3)$  Lie group, the pose is then updated as:

$$E'_{CW} = \exp(\mu) E_{CW} \quad (3.32)$$

### 3.1.5 Priors

In order to improve the method, we include regularization and smooth priors in the optimization. We call them *tracking priors* and they allow us to obtain better 3D error and serve to smooth the solutions of pose and shape along the sequence.

After a revision of the related works, we found out that the more suitable priors for this tracking problem are temporal and shape smoothness, as they are easily adapted to a least squares minimization scheme in the E-step.

Other more complex priors were considered, such as the isometric prior. We discarded them as they generate non-convex costs which don't fit well in our EM approach. Isometry is also a strong prior that cannot be used for some objects that undergo elastic deformations. We refer the readers to Appendix B for more details.

The consequence of including the priors is the modification of the error function to be minimized. Instead of relying only on the 2D reprojection error, the cost includes the following terms:

$$E = E_{data} + \rho_{temp}E_{temp} + \rho_{shape}E_{shape} \quad (3.33)$$

where  $E_{data}$  is the 2D reprojection error described in Eq.(3.8),  $E_{temp}$  imposes temporal smoothness and  $E_{shape}$  is an extra restriction of the shape, to not deviate its shape from the original one (spatial smoothness).

Now, each of the terms is explicitly defined:

$$E_{data} = \sum_{i=1}^P \|U'_i - proj(X'_i)\| \quad (3.34)$$

$$E_{temp} = \sum_{i=1}^P \|X'_i(f) - X'_i(f-1)\| \quad (3.35)$$

$$E_{shape} = \sum_{i=1}^P \|X_i(f) - \hat{X}_i(f)\| \quad (3.36)$$

where  $\rho_{temp}$  represents the weight for temporal smoothness and  $\rho_{shape}$  is the weight for shape smoothness.

In  $E_{temp}$  imposes a temporal dependence between the current 3D shape with respect to previous time  $X_i(f-1)$ .

The expression of  $E_{shape}$  forces the current shape  $X_i(f)$  to be closer to the shape  $\hat{X}_i$ , obtained by predicting each node of the mesh using its neighbors:

$$\hat{X}_i(f) = \sum_{k,m,n \in \mathbb{N}(\hat{X}_i)} \alpha_i X_k + \beta_i X_l + \gamma_i X_m \quad (3.37)$$



where  $\alpha_i$ ,  $\beta_i$  and  $\gamma_i$  are the barycentric coordinates of vertex  $i$  in the mesh with respect to three neighboring vertices  $X_k$ ,  $X_l$  and  $X_m$ . Note that the barycentric coordinates are computed using the reference shape. This term imposes spatial smoothness by penalizing shapes with points that do not agree with their neighbors. Eq. (3.33) can be optimized in the E-step with linear least squares. We refer the reader to Appendix B for more details.

### 3.1.6 Tracking recovery

This procedure is inherited from the original PTAM tracking algorithm, which is described in the active search algorithm presented in [Williams et al., 2007]. The algorithm was introduced before PTAM was released, so it was integrated and tested in monoSLAM [Davison et al., 2007].

In the original PTAM method, tracking recovery mode is outside the normal operation. It is applied when the underlying map is considered good (i.e. it is not rejected by the Sparse Bundle Adjustment (SBA) implemented in the mapping thread), but the tracking is lost in several consecutive frames.

There are very few approaches that implement recovery strategies for live tracking on NRSfM / SfT systems, presumably because a good tracking is assumed, and as a consequence the matching is not considered a problem. Although we consider tracking recovery procedures are important in real tracking systems to maintain them stable.

Tracking performance is assessed from the amount of tracks correctly matched among the visible ones. When the number of tracks detected with respect to the ones found in the image is low for a certain number of consecutive frames (3 by default), a recovering procedure is started to find out a pose correction that better match the points.

When a tracking loss is detected, the active search algorithm is started. It consists of estimating a 2D rotation between the latest correct frame feature patches (stored on the map) and the current frame feature patches. The estimated 2D rotation is then upgraded to a 3D rotation and the current pose is updated to carry on with the tracking estimations. If this procedure is not successful in the current frame, it is repeated in the following frames until a successful rotation gets enough matches. As stated, this process assumes rigid scenarios. If a successful rotation has been found, normal tracking follows this recovery procedure.

However, the active search recovery must be revisited for the non-rigid scenarios. The closed-form of estimating the pose makes the shape of the object is also affected over the frames, as the deformation coefficient estimation step relies on having a good estimation of the pose. If the rotation abruptly changes from frame to frame, the shape will abruptly change to adapt it to this rotation change. In the results section 3.2.2.3.4 we study the impact of changing or not the shape during the re-localization scenario during the active search.

This leads to abrupt shape changes during the frames where a re-localization is done, as the changes are high and non-linear in all the parameters, trying to match quickly the highest number of features as possible. Therefore, the application of time smoothness priors in this case is not useful, as it lowers the convergence speed, having in mind that the main target of a recovery procedure is to get back to a normal state as soon as possible.

## 3.2 Results

Firstly, a general overview of the performance metrics is described, after that a revision of the different datasets used for the assessment of the algorithm are presented, as well as a detailed report of the results for each of the sequences.

### 3.2.1 Performance metrics

The metric used to evaluate the performance of our proposal for different test sequences (motion captured or synthetically generated) is depicted in this section. After a sequence is processed by the algorithm, the output results are post-processed to compare the performance of the algorithm with the ground truth. The way that the motion capture and synthetically generated sequences are adapted to serve as inputs of the system is also explained.

The fundamental equations to evaluate the error of the output estimations are the following ones:

2D error:

$$2D\ err(px) = \frac{1}{F} \sum_{f=1}^F \frac{\sqrt{\sum_p \|x_{est}(f, p) - x_{gt}(f, p)\|_2^2}}{\sqrt{\sum_p \|x_{gt}(f, p)\|_2^2}} max(x_{gt}) \quad (3.38)$$

where  $F$  is the number of frames of the sequences,  $P$  is the number of points,  $x_{est} = (u_{est}, v_{est})$  are the re-projected features on the image and the  $x_{gt} = (u_{gt}, v_{gt})$  the ground truth in image coordinates.

3D error:

$$3D\ err(\%) = \frac{1}{F} \sum_{f=1}^F \frac{\sqrt{\sum_p \|S_{est}(f, p) - S_{gt}(f, p)\|_2^2}}{\sqrt{\sum_p \|S_{gt}(f, p)\|_2^2}} * 100 \quad (3.39)$$

being  $S_{est} = (X_{est}, Y_{est}, Z_{est})$  the estimated 3D shape, and  $S_{gt} = (X_{gt}, Y_{gt}, Z_{gt})$  the ground truth shape.

translation error:

$$t \text{ error } (\%) = \frac{1}{F} \frac{\sqrt{\sum_f \|T_{est}(f) - T_{gt}(f)\|_2^2}}{\sqrt{\sum_f \|T_{gt}(f)\|_2^2}} * 100 \quad (3.40)$$

in which  $T_{est} = (t_{est,x}, t_{est,y}, t_{est,z})$  is the estimated translation vector and  $T_{gt} = (t_{gt,x}, t_{gt,y}, t_{gt,z})$  the ground truth camera translation vector.

rotation error:

$$rot \text{ error } (\%) = \frac{1}{F} \frac{\sqrt{\sum_f \|\Theta_{est}(f) - \Theta_{gt}(f)\|^2}}{\sqrt{\sum_f \|\Theta_{gt}(f)\|^2}} * 100 \quad (3.41)$$

where the  $\Theta_{est} = (\theta_{est,x}, \theta_{est,y}, \theta_{est,z})$  corresponds to the estimated 3 rotation angles, and  $\Theta_{gt} = (\theta_{gt,x}, \theta_{gt,y}, \theta_{gt,z})$  the ground truth rotations.

The 2D reprojection error and the 3D reconstruction error were used in [Paladini et al., 2010]. Following a similar criteria, two new equations are applied for the rest of the state vector parameters such as the rotation angles and the translation vector coordinates.

Even though global errors are calculated this way, those expressions do not work properly for analysis over time, in this case a per frame error is more useful than its mean.

In order to obtain the 3D error, before applying Eq. 3.39, a previous Procrustes analysis [Gower and Dijksterhuis, 2004] is done to rotate and scale the shapes for a correct comparison, and also to do the comparison in the same conditions as other works on the state-of-the-art as [Paladini et al., 2009, Paladini et al., 2010]. Unless it was explicitly indicated, Procrustes shape alignment is applied to compare the shapes.

The number of bases considered for all the experiments are at most  $K = 15$ , unless indicated. It is usually considered that the optimum number of bases is the one that allocates at least the 85% of the total *deformation energy* obtained from the Principal Component Analysis (PCA) decomposition of the shape matrix. Not all the state-of-the-art methods reaches this ideal rank, and a high number of bases could lead to overfitting when dealing to real data, not to mention the memory and computational costs for handling a large number of bases. These are the main reasons why a fixed value (unless specified) of bases is set in this Thesis.

In order to handle the data from motion captured datasets, which involves the absence of any kind of visual information, a special version of the tracking thread was developed, to accept as input text files containing the point projections for each frame, the 3D bases, the pose initialization, the visibility masks, etc.

- The ground truth points are projected using the perspective camera as shown in Eq.(3.3), choosing a point of view far enough to see all the sequence points in an approximately frontal view for all the frames.

- Afterwards, the mean is taken out from the 3D and then they are rearranged so as to have a  $F \times 3P$  shape, being  $F$  the number of frames and frames and  $P$  the number of points.

The mean 3D shape is also saved to disk, to be loaded afterwards by the tracking algorithm.

$$S - S_0 = \begin{pmatrix} x_{1,1} & y_{1,1} & z_{1,1} & \cdots & x_{1,P} & y_{1,P} & z_{1,P} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{F,1} & y_{F,1} & z_{F,1} & \cdots & x_{F,P} & y_{F,P} & z_{F,P} \end{pmatrix} = UDV \quad (3.42)$$

PCA is applied to this matrix and the most relevant components are taken, which are 15, as indicated before. For these experiments 15 bases are more than enough to cover more than the 85% of the *deformation energy*, which is computed from the trace of matrix  $D$  as follows:

$$D_{normalized} = \frac{diag(D)}{\sum diag(D)} \quad (3.43)$$

$$D_{cumulated}[i] = \sum_{j=0}^i D_{normalized}[j] \forall i \quad (3.44)$$

In each of the analyzed sequences, (unless is derived from an other), the *deformation energy* analysis will be given at the beginning of each section.

In order to better clarify this point, no Procrustes is applied before the factorization. From the SVD factorization, the approximation to the metric projection matrix chosen is the  $\sqrt{D}$ , so the coefficients ( $L = U\sqrt{D}$ ) and the bases ( $B = \sqrt{D}V$ ) are obtained. As  $D$  is diagonal it does not need to be transposed. After being reduced to the first  $K$  components, the *vectorization* operation is reverted, to have the bases in a  $3K \times P$  format, ready to be loaded as a model.

To perform a thorough comparison and simulate real tracking conditions for these datasets, an increasing noise strength,  $\sigma = [0, 1, 2, 3, 4]$  and outlier percentage,  $outl = [0, 5, 10, 20, 30, 40]\%$  is added to the set of points, as similarly done by [Moreno-Noguer and Porta, 2011]. With this setup, an initial evaluation of the behavior of the algorithm robustness to these conditions is performed. The introduced noise follows a normal distribution  $N(0, \sigma^2)$ . The outliers are introduced in random points in the percentage indicated, and for those points both spacial directions are randomly deviated 20 pixels. It is also introduced the visibility as a variable to analyze algorithm robustness. To do that a set of files including randomly distributed visibility masks in an increasing percentage are generated. These are very important parameters to take into account for the rendered experiments.

All the experiments were computed using a i7 laptop with 8 virtual core processor at 2.4 GHz, with 8GB RAM. The algorithm is implemented in c++, runs on Ubuntu Linux, using CGAL, OpenMP, OpenCV and PTAM derived libraries. The experiments for , [Torresani et al., 2008], [Paladini et al., 2009], [Paladini et al., 2010] and [Gotardo and Martinez, 2011] are run with the default parameters set in the examples of the corresponding papers. We set the motion movement constant,  $\beta$ , to 0.9.

In order to be fair, as the presented approach is model based, it should be compared with other model-based / SfT approaches of the state of the art, like [Moreno-Noguer et al., 2009], [Chhatkuli et al., 2014b] and other approaches based on templates. In the results comparison tables we show this parameter for fair comparison.

### 3.2.2 Sequences

A general overview of the algorithm is presented on the easier sequences, getting higher analysis over more difficult ones. On motion captured datasets we start evaluating the tracking assuming the matching is perfect, after that, the results are re-evaluated for tracking degenerations such as visibility, noise, outliers, and incomplete bases. Neither priors nor recovery are applicable.

For real images, all the addressed conditions have to be met at the same time, as a real matching algorithm must be handled for each frame, so there is no need to evaluate them separately.

#### 3.2.2.1 CMUface Sequence

This motion captured sequence represents a moving head turning and talking. The CMU-face sequence was proposed by [Paladini et al., 2010]. It contains 40 points in 316 frames projected in a 640x480 image.

It is checked by using 15 bases, which is corresponded to a *deformation energy* of 95.8%, complying with the assumption of a standard PCA decomposition.

##### 3.2.2.1.1 Performance evaluation based on perfect matching

First of all, some screenshots of the sequence are shown in Fig. 3.3. The white points correspond to the input tracks and the red points to the projection of the estimated 3D shape for the current frame. As there are no imperfections added to the tracking, it is hard to find the red points most of the times. The first iterations of the algorithm are enough to fit the shape and the pose of the current frame.

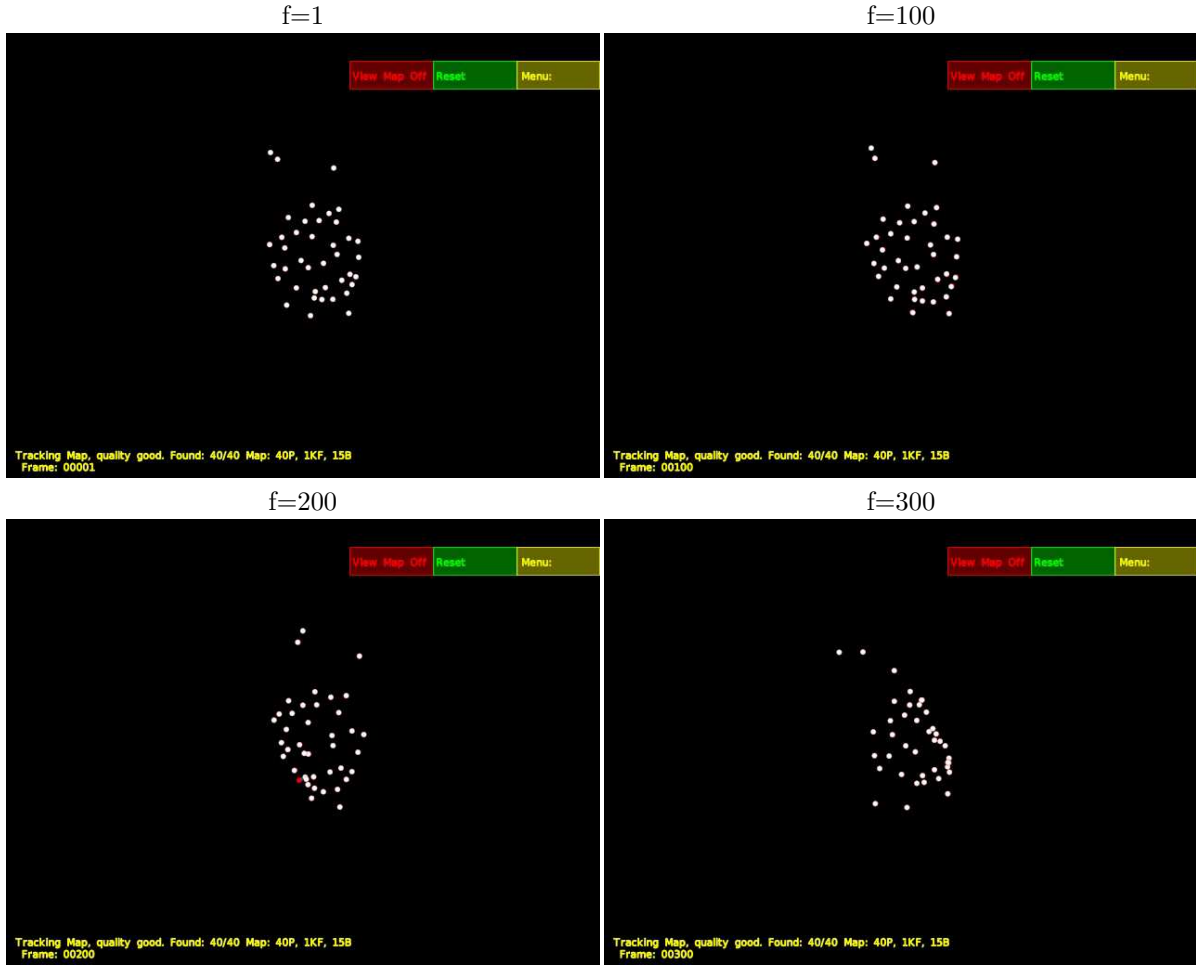


Figure 3.3: CMUface sequence snapshots for certain frames

Some 3D reconstructions compared to the ground truth for some frames are depicted in Fig. 3.4. In this case, there are no noise and outliers present on the tracking. There is also no additional relative movement outside the modeled one between the object and the camera. The ground truth points are represented in blue and the red circles are the reconstructions. It can be seen, there are very few occasions where the blue points get out of the red circles on the reconstructions for the presented frames. This dispersion is greater as the depth gets further.

The relative movement between the camera and the object can be seen in Fig. 3.5. It can be deduced that, for this case, the relative movement between them is insignificant, or the reference system is joint. It contrasts to what it can be seen on Fig. 3.3 that for frames #200 and #300 there exist some rotation of the head.

The absence of relative motion can be explained as even though pose change do not appear small, when the model was generated, no Procrustes was applied before generating the bases. As consequence, the inner rotations were modeled as part of the deformations in the bases. If Procrustes had been applied, rotations would have been estimated. Therefore, the rotation and translation estimation can not be compared exactly against

the ground truth. However, considering deviations are small we have carried out the evaluation of these parameters.

The performance of the tracking in terms of reprojection and 3D reconstruction is shown in Fig. 3.6. It can be seen, for ideal tracking conditions, that the algorithm presents low 2D and 3D errors for this sequence. The reasons could be that the complexity of the sequence is not very high as the only deformation is located in the mouth when the person is talking, remaining the rest of the parts of the sequence rigid, although the different depths of the face makes the perspective reconstruction challenging because it is affected by this parameter.

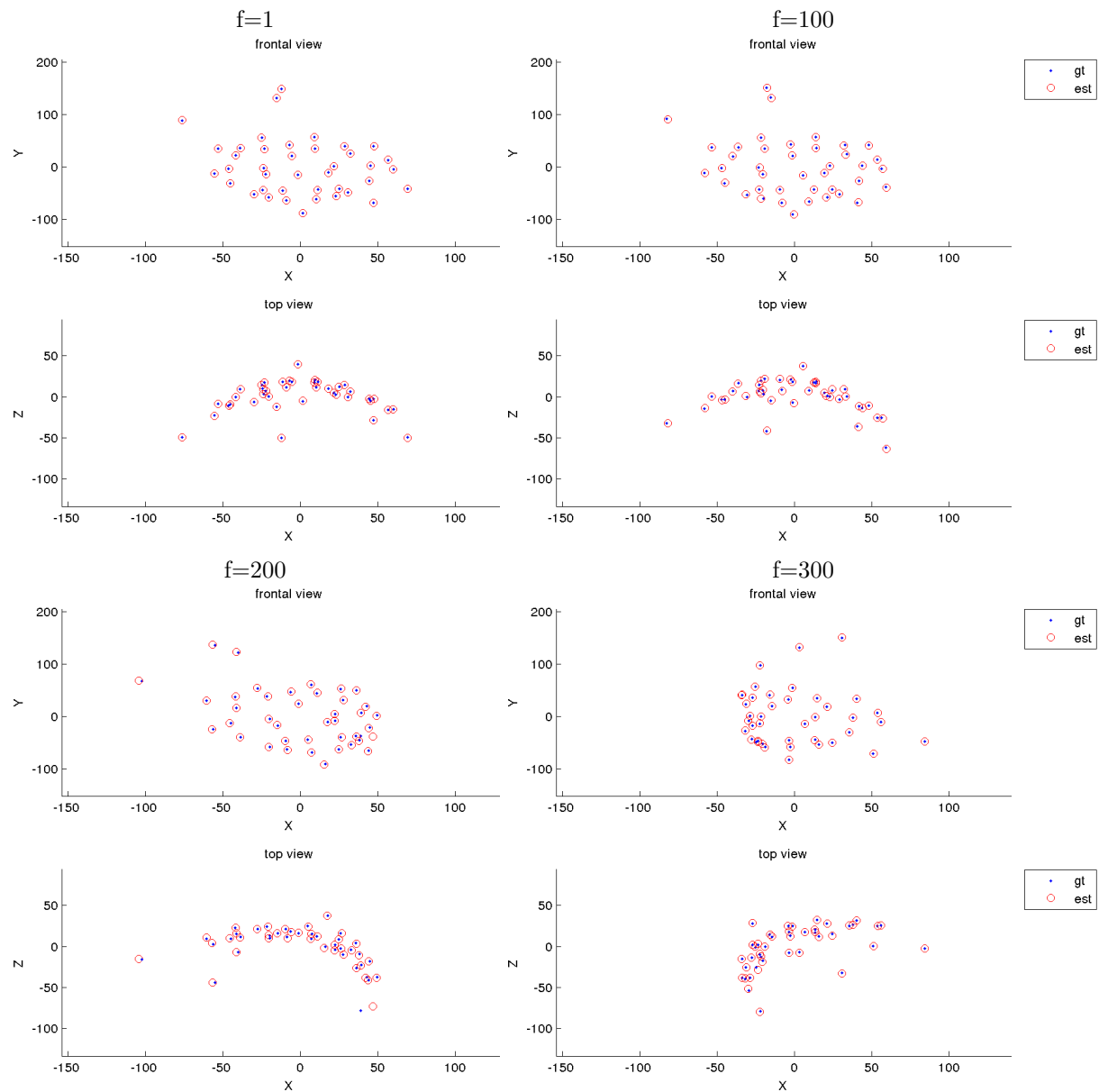


Figure 3.4: CMUface sequence reconstruction examples without any tracking degradation effects. In blue the ground truth and red the estimated shape.

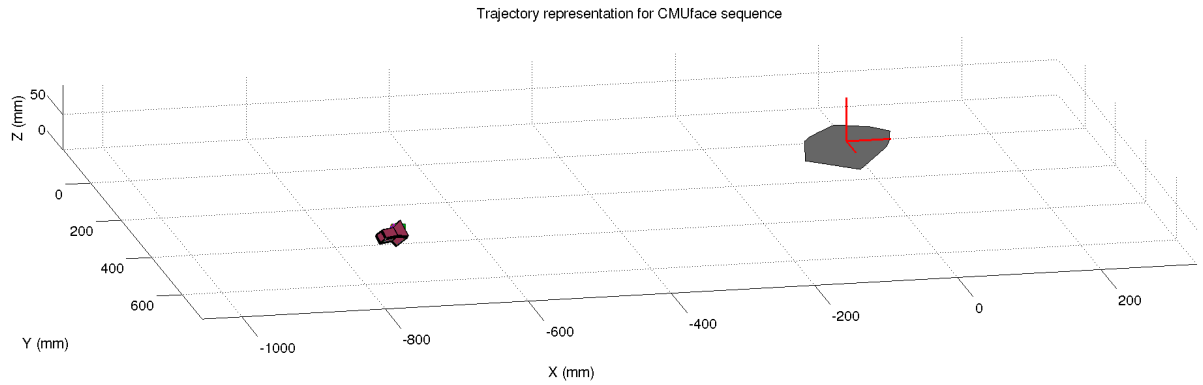


Figure 3.5: Trajectory estimation for CMUface sequence when no tracking degradation is added. The relative movement between the camera and the object is barely static, as it can be seen in the reconstruction.

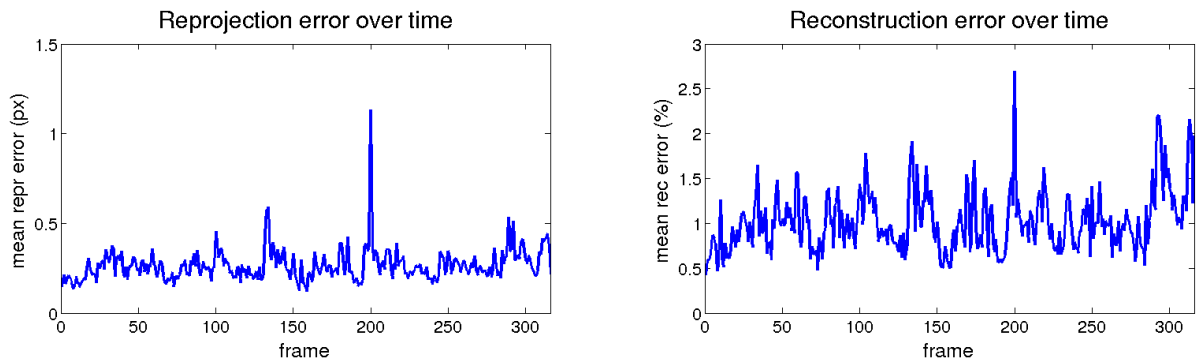


Figure 3.6: Representation of the errors over time for CMUface sequence. At the left the 2D reprojection error and at the right the 3D reconstruction error.



### 3.2.2.1.2 Performance evaluation based on visibility degradation

After the analysis with a perfect matching along the sequence, condition that most state-of-the-art works related to NRSfM / SfT assume, some degradation is included to evaluate its robustness against different real factors. In this section the visibility is evaluated. A visibility mask is randomly generated for each of the tracks of each of the frames, in a given percentage and then the whole sequence is evaluated.

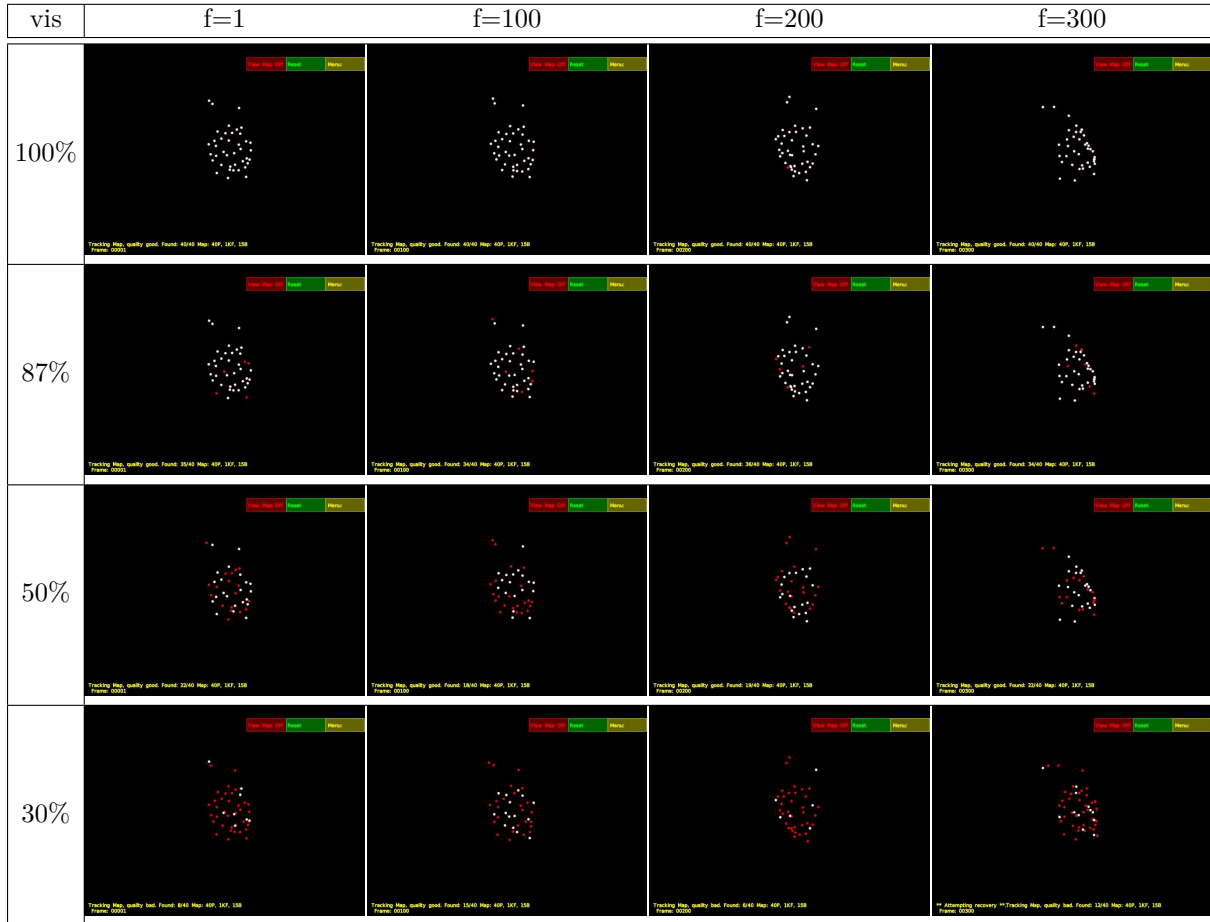


Figure 3.7: CMUface sequence screenshots for visibility experiments

Some screenshots of the experiments are shown in Fig. 3.7. As in Fig. 3.3, the white points are the visible points, the hidden points are not rendered, and the red points are the projections of the estimated 3D model computed for the current frame.

The results in Fig. 3.7 are shown in a grid, ordered in frames per columns and visibility per rows. It can be seen that for the same frame, the reconstructions are similar for more of the visibility degradation cases but for the lowest visibility cases precision is compromised, as expected.

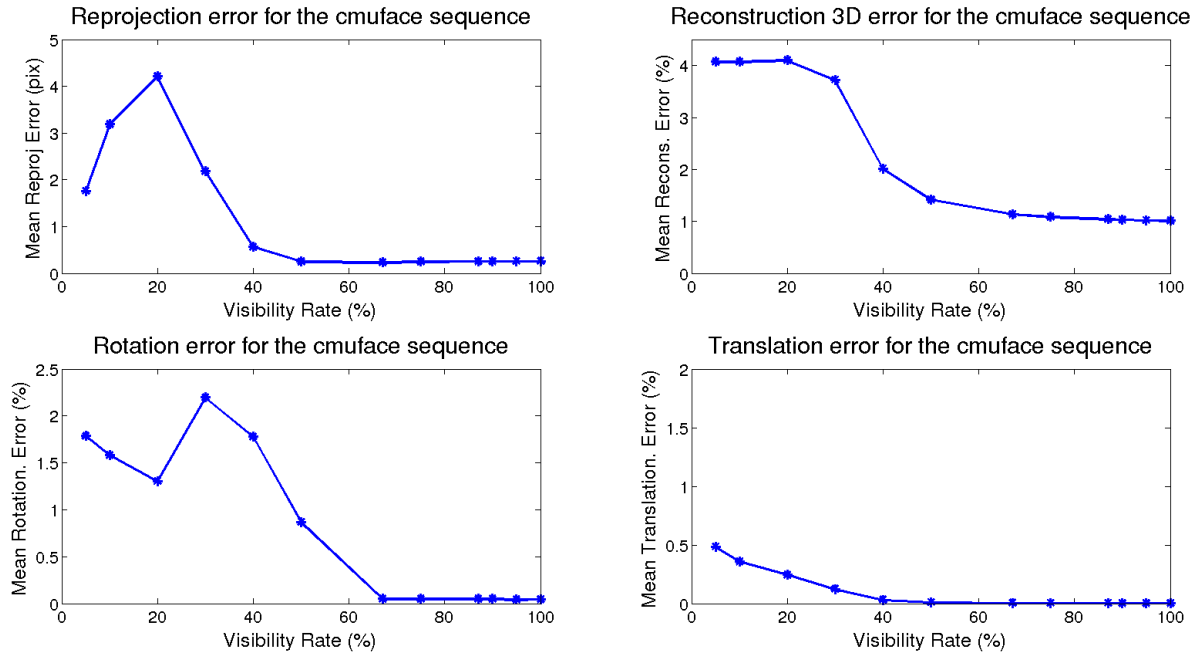


Figure 3.8: CMUface sequence error results for visibility degradation on tracking. Left top subfigure depicts the variation of the reprojection error. Top right the same with the 3D reconstruction error. Bottom left shows the rotation error evolution and in the bottom right the translation error.

Some results on a range from 5% to 100% of visibility are presented in Fig. 3.8. The evolution of the errors give an idea of the minimum percentage of points visible to perform correct estimations. It must be also noted that this sequence only has 45 points, which limits the analysis range until 20% (9 points). Lower values would yield no significant values or wrong estimations, as the number of points to perform the estimation are not significant, as it can be seen in the reprojection and rotation errors below the 20%.

The experiments performed here were repeated 10 times, so as to refine the results and provide averaged values. In the current case, it can be seen that a minimum reprojection error can be set until 50% visibility which means at least 20 points must be visible for this sequence, whereas for an accurate 3D reconstruction, the minimum set of available points should be 67%, which means for this sequence, at least 27. It must be also remarked that the non-visible points are randomly generated, which means that real occlusions due to face turns similar to full visibility are not correctly modelled.

Finally, for a visibility rate higher than 67%, the results are similar to full visibility. The mean reprojection error is about 0.2 pixels and the reconstruction error about 1%, which are the value that yields the perfect tracking, as seen in Fig. 3.6. The translation and rotation errors are about 0%.

The big impact on the reconstruction will be given when certain areas on an image get occluded (face turning simulation), so not a proper model on that area would be estimated and the 3D reconstruction performance would be significantly lowered.

### 3.2.2.1.3 Performance evaluation based on noise and outliers

Other causes of degradation on the tracking are the noise on the measurements and the presence of the outliers on the matching process that are not removed by the rejection algorithms. Hereafter, the robustness analysis against these factors is performed.

How the sequence is prepared to perform the experiments was explained in 3.2.1. In this section, random noise and outliers are added to the tracks in the indicated parameter:  $\sigma$  for noise std deviation and  $o$  for outlier percentage.

Some, screenshots of the sequence are shown in Fig. 3.9, in which the white points are the input tracks (including noise and outliers) and red points are the 3D estimation for projected on the image the current frame.

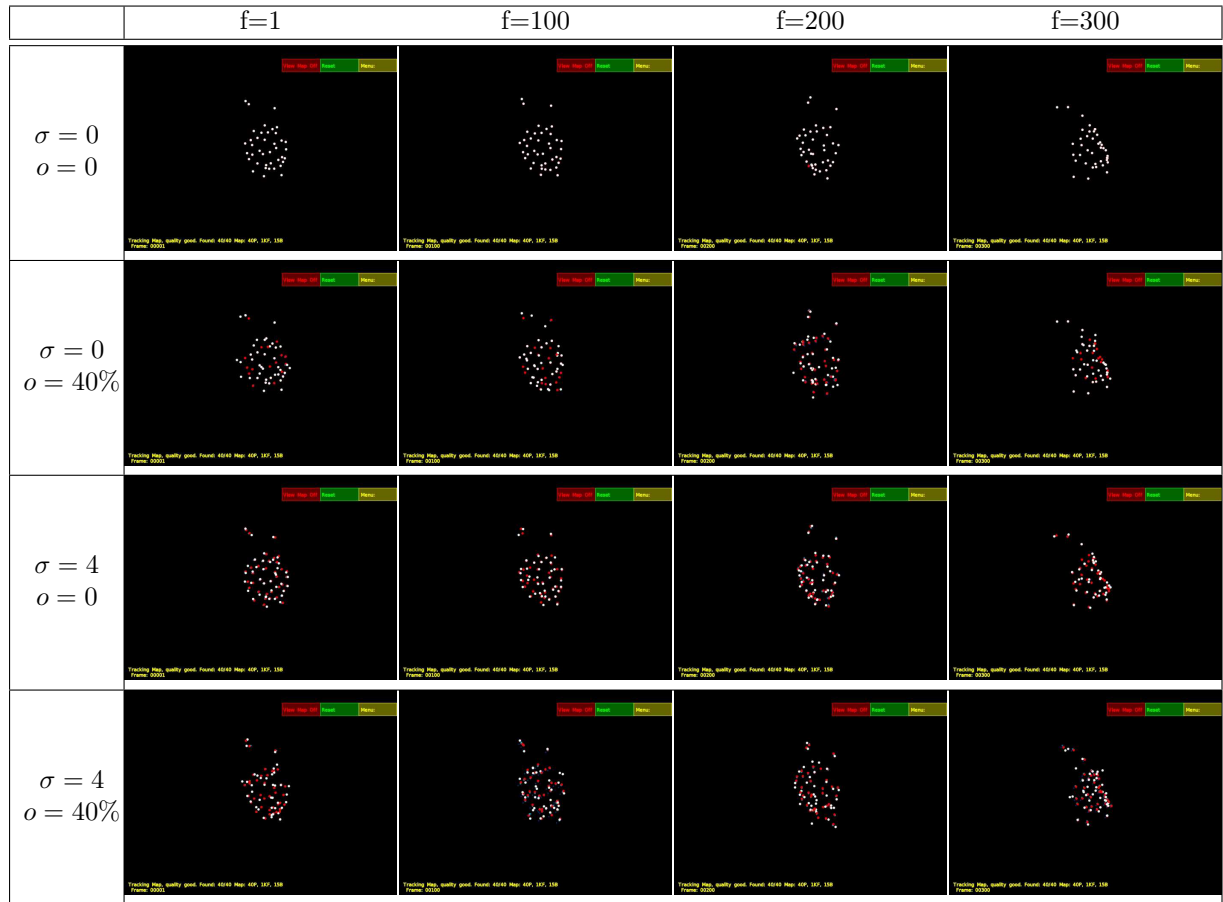


Figure 3.9: CMUface sequence screenshots for noise and outlier experiments

The summary of the results after the processing is depicted in Fig. 3.10. Several parameters combination are evaluated in a representative range to get conclusions.

It can be observed that the estimations are stable with an outlier rate of the 20% for both the reprojection and 3D reconstruction error for the least noisy measurements. For higher values the error grows up exponentially.

The effect of adding noise to the measurement, when no outliers are added makes the error increment almost linear, as the separation between traces is almost maintained. As the outlier percentage grows up, the separation between traces is wider so the influence of these two parameters is mixed in the estimation.

With respect to the rotation subfigure it can be seen that it is a bit more mixed up than the rest of the figures. Even though the experiment was repeated 10 times, more repetitions would be needed to get smooth results. Even so, a similar trend to the rest of the error traces can be seen. Regarding the translation, the trend is similar to the reprojection and reconstruction errors, although it presents a very low value of error.

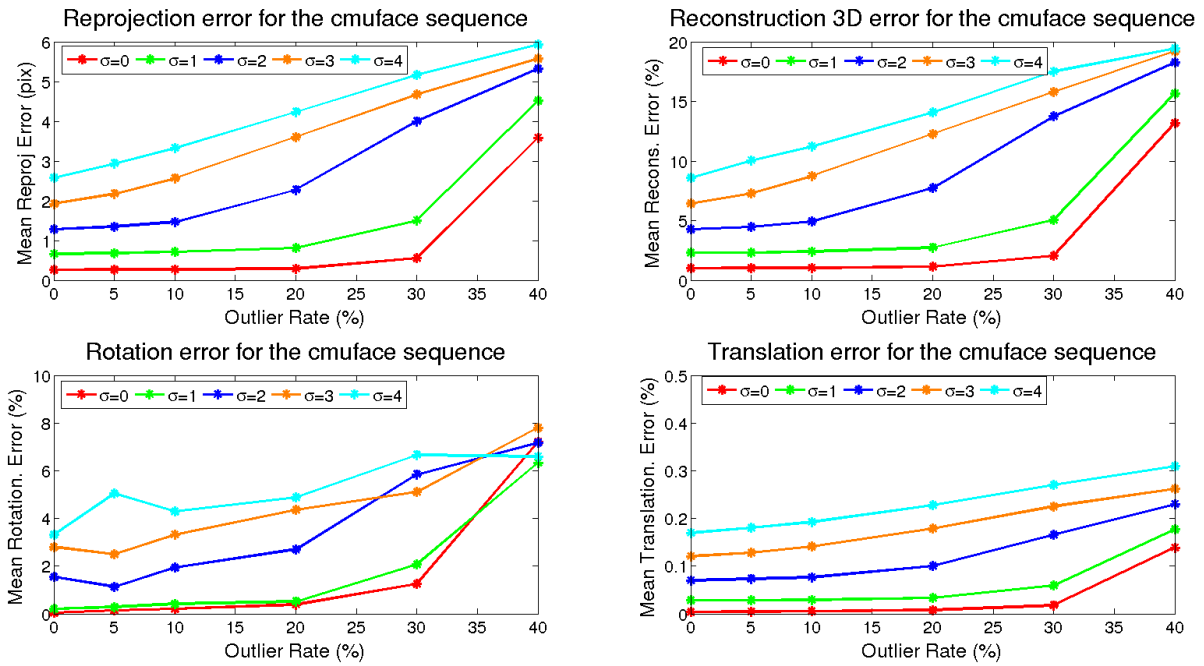


Figure 3.10: Noise and outliers results for CMUface sequence. Left-top sub-figure depicts the variation with the percentage of outlier points of the reprojection error, each trace corresponds to a different noise  $\sigma$  level. Top-right shows the same with the 3D reconstruction error. Bottom-left shows the dependence of the rotation error and, in the bottom-right with the translation error.

#### 3.2.2.1.4 Performance evaluation based on the number of bases

A fixed number of bases has been taken for all the experiments ( $K = 15$ ), although it is also interesting an analysis using a different number of bases to observe its trend.

To carry out the proposed tests, we use the tracks without degradation (perfect tracking) and 7, 15 and 30 bases are used.

With this setup, the results are shown in Table 3.1.

#Bases( $K$ )	2D error (pix)	3D error(%)	rot. error(%)	trans. error(%)
7	0.59	1.67	0.45	0.0075
15	0.26	1.01	0.04	0.0038
30	0.14	0.71	0.0251	0.0022

Table 3.1: CMUface sequence error comparison with the number of bases, for 7, 15 and 30.

As expected, with perfect tracks, as the bases increases the error decreases for all the parameters evaluated. In addition, the processing time improves as the number of bases are reduced, as well as the memory used.

To confirm the results depicted in the Table 3.1, the Fig. 3.11 illustrates the error analysis over time.

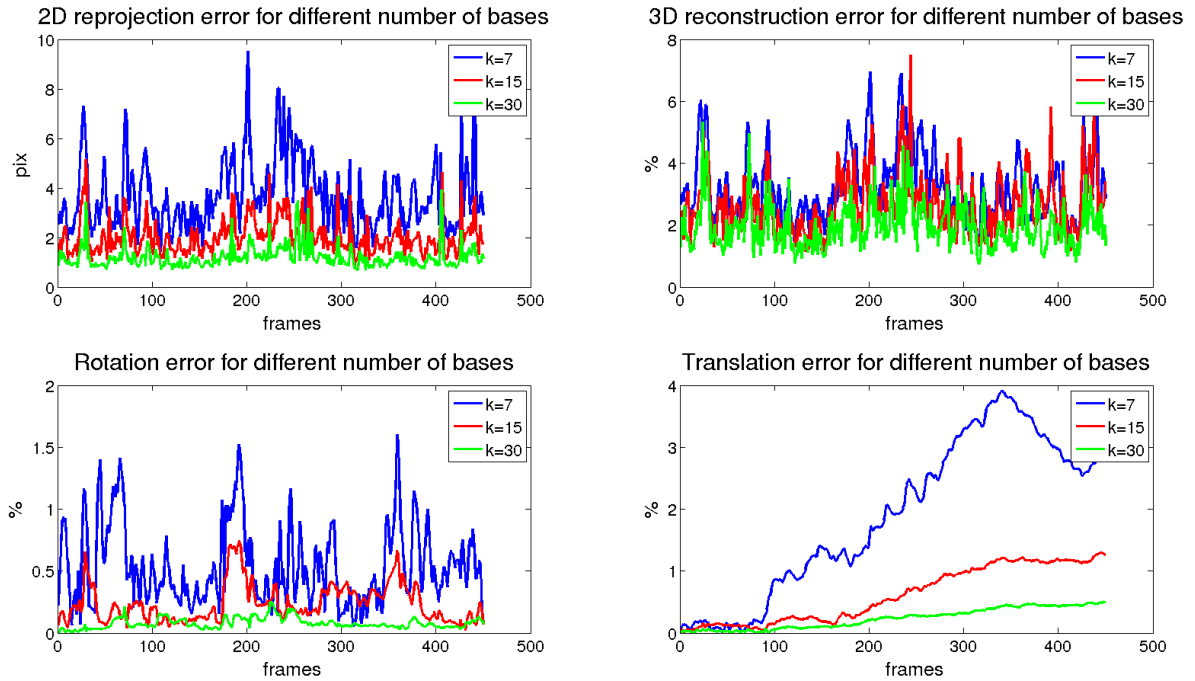


Figure 3.11: CMUFace sequence Error over time varying the number of bases. The four error figures are shown. Top left shows the 2D reprojection error. The top-right shows the 3D reconstruction error. The bottom left shows the rotation error and the bottom right shows the translation error

### 3.2.2.1.5 Comparison with other methods of the state-of-the-art

In Table 3.2 a summary of results against other state-of-the-art algorithms for sparse proposal is shown, depicting total and per frame processing time, 3D accuracy, 2D error, T. error, rot. error and rank. Our proposal gets the best 3D accuracy and reprojection error. The results are obtaining for perfect tracking in order to keep the same conditions that our competitors.

The methods were run by using their own source code, and tested on the same hardware. The type of perspective projection were selected according to the requirement of each method.

If the algorithm is not able of reaching the maximum rank up to 15, the maximum reached on the experiment is indicated.

Method	2D err(px)	3D err(%)	rank	Tproc (fps)	model	proc type
[Paladini et al., 2010]	1.06	3.18	8 (max)	14 min (0.37)	auto	seq
[Gotardo and Martinez, 2011]	0.6	3.19	-	43 seg (7.35)	auto	batch
[Torresani et al., 2008]	6.42 / 32.39	9.9 / 56.06	5 / 15	78 / 606 seg (4.05/0.52)	auto	batch
[Paladini et al., 2009]	1.06	2.43	12	13 seg (24.3)	auto	batch
Our method	<b>0.26</b>	<b>1.01</b>	15	<b>4.5 seg (69.78)</b>	priory	seq

Table 3.2: CMUface sequence results summary. Some results comparing different state-of-the-art methods against the presented one is depicted

This sequence is ideal to a first test as it contains few points and results are fast to obtain in order to be compared with the state-of-the-art algorithms publicly available. Deformations presented on the sequence are not very challenging as they are only due to the mouth opening and closing, but they are combined with translations and rotations of the face over time.

It could seem unfair comparing a model-based / SfT approach, as is our tracking proposal, with pure NRSfM algorithms, but similar comparisons are performed on state-of-the-art papers that tackle this problem.

### 3.2.2.2 Flag Sequence

In this sequence a motion captured flag is bending with the wind. This sequence was introduced in [White et al., 2007]. Deformations are significant which makes this a challenging dataset. Moreover, the amount of points in the sequence (540 in 450 frames projected in a 640x480 image) is significantly higher than in other data sets, which implies a challenge to algorithms when matrix factorization takes place. In order to solve this problem some works are tested with a reduced amount of points.

The procedure to extract the input data to our algorithm is the same as the one used for the CMUface sequence.

Following a similar guideline as in the previous section, some screenshots of the sequence are firstly shown in Fig. 3.12. It can be seen that the reconstructions are not as exact as the ones in the CMUface sequence, although for most of the tracks the error is low. The *deformation energy* kept by the first 15 bases ( $K = 15$ ) is this time 87.33%, which follows the imposed criteria of keeping the 85% of the *deformation energy*.

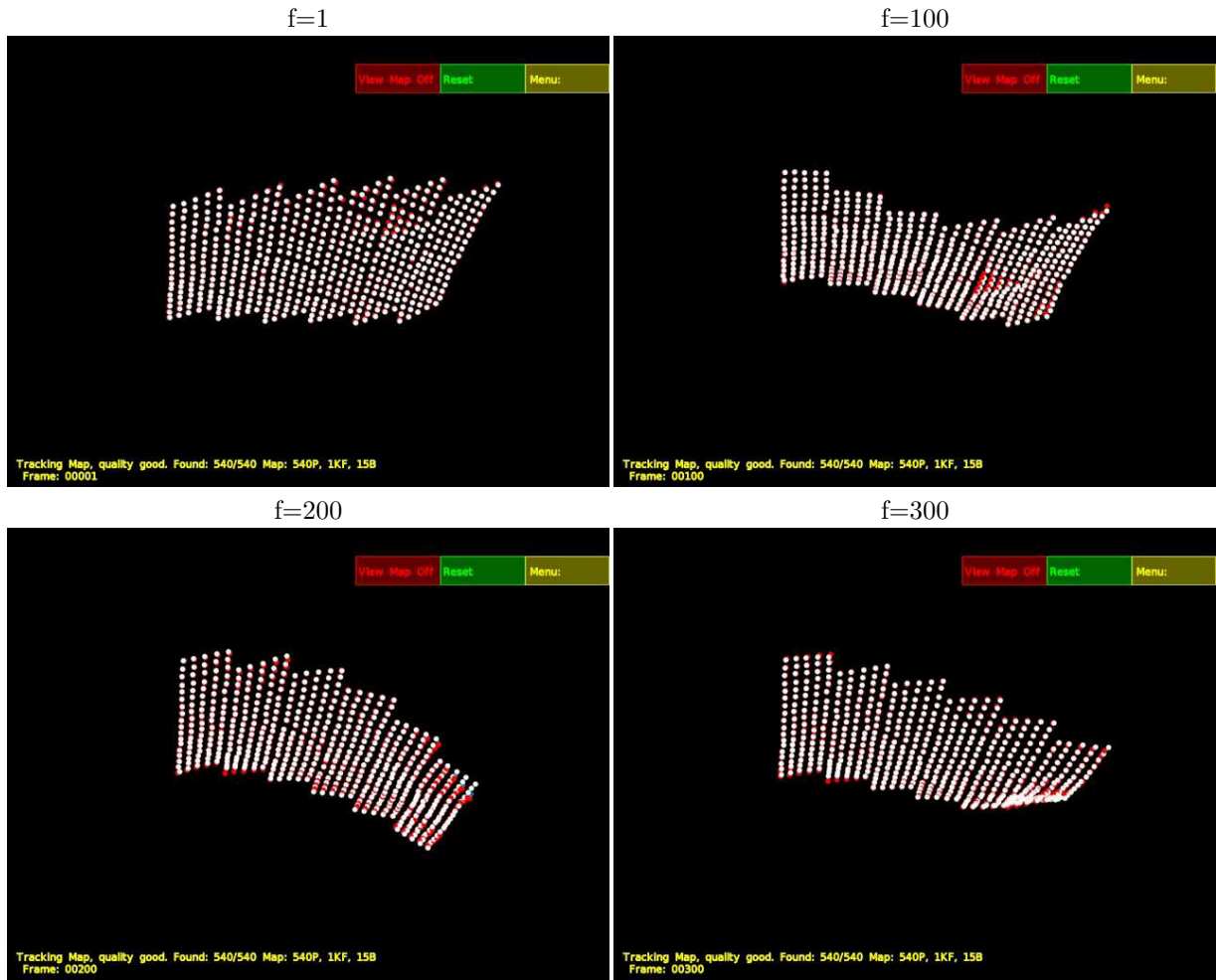


Figure 3.12: Flag sequence snapshots for certain frames

### 3.2.2.2.1 Performance evaluation based on perfect matching

The 3D reconstruction results without any degradation on the matching are shown in Fig. 3.13. They are compared with the ground truth, which are represented in blue whilst the reconstruction result is represented in red. Two views are shown to better view the 3D reconstruction. It is important to remark that the shapes are aligned using Procrustes to have a proper comparison, as other related works do. For most of the frames, the reconstruction seems very accurate. The one with more error is seen in frame #200, on one of the corners, which corresponds to one of the most moving cases, which has not been properly modeled.

The trajectory reconstruction is shown in Fig. 3.14. It can be seen that the relative movement between the camera and the object is almost inexistent, or, as indicated in the previous sequence, it is integrated on the bases on the factorization.

The reprojection and 3D reconstruction errors over time are presented in Fig. 3.15. This time the reprojection and the 3D reconstruction error are higher than for the CM-Uface sequence. Due to the 15 selected bases represent a *deformation energy* lower than

for the former sequence. For a perfect matching, increasing the number of bases would minimize these errors. However, this trend is not met when real matching features are used.

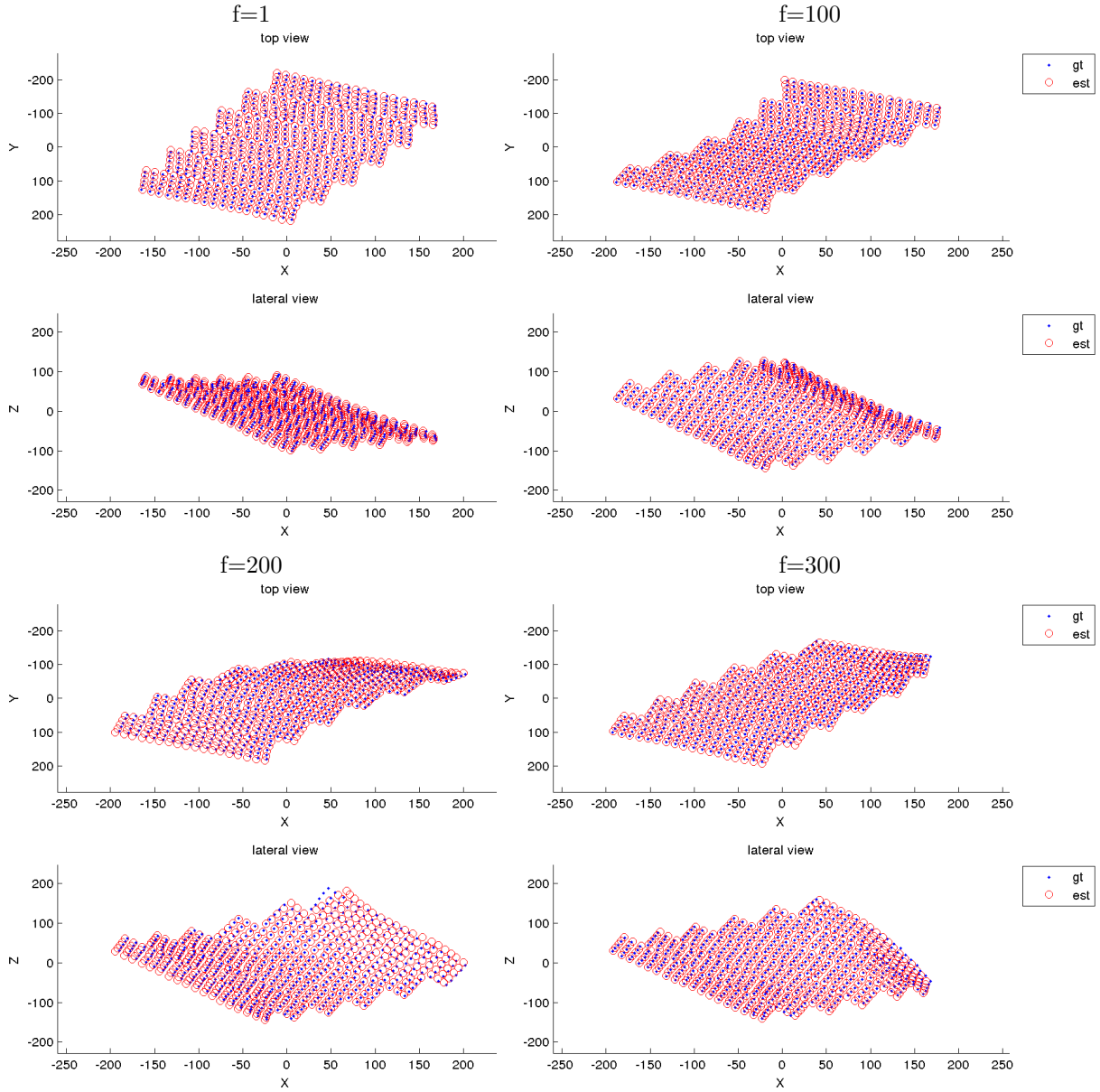


Figure 3.13: Flag sequence reconstruction examples without any tracking degradation effects



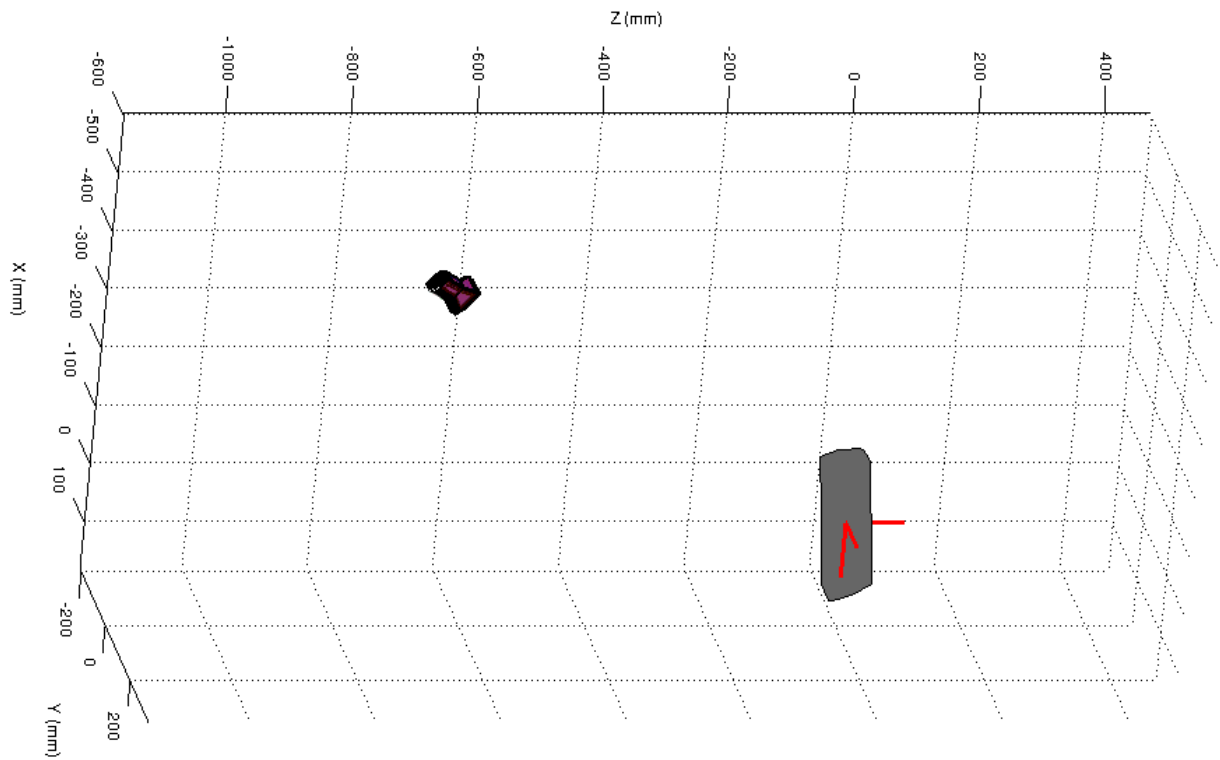


Figure 3.14: Trajectory estimation for Flag sequence without tracking degradation. The camera is barely static, as it can be seen in the reconstruction.

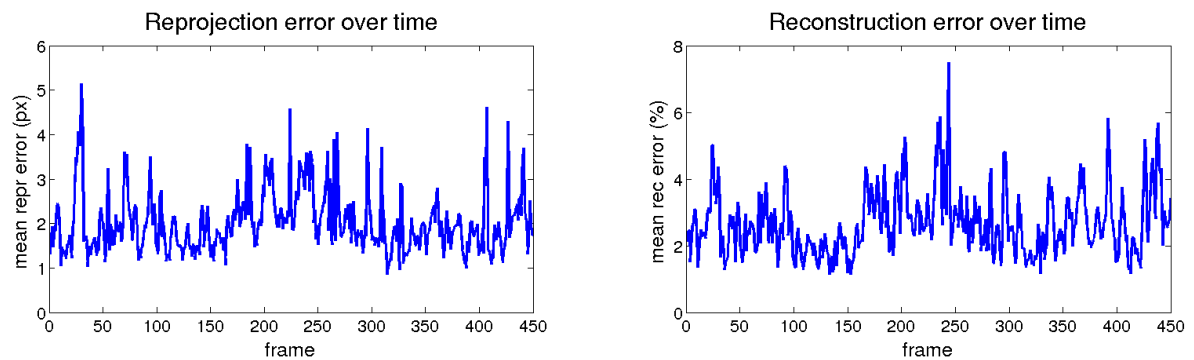


Figure 3.15: Flag sequence representation of the errors over time. At the left the reprojection error and at the right the 3D reconstruction error

### 3.2.2.2.2 Performance evaluation based on visibility degradation

As previously done for the CMUface sequence, the not detection of features is simulated by generating visibility masks and loading them from a file. These masks are generated according to a percentage, depending on the quality of the matching to be simulated for each frame. The visibility distribution of the masks for each point and among frames is completely random.

For this sequence, some of the screenshots of the different trials with variations on the visibility percentage per rows can be seen in Fig. 3.16.

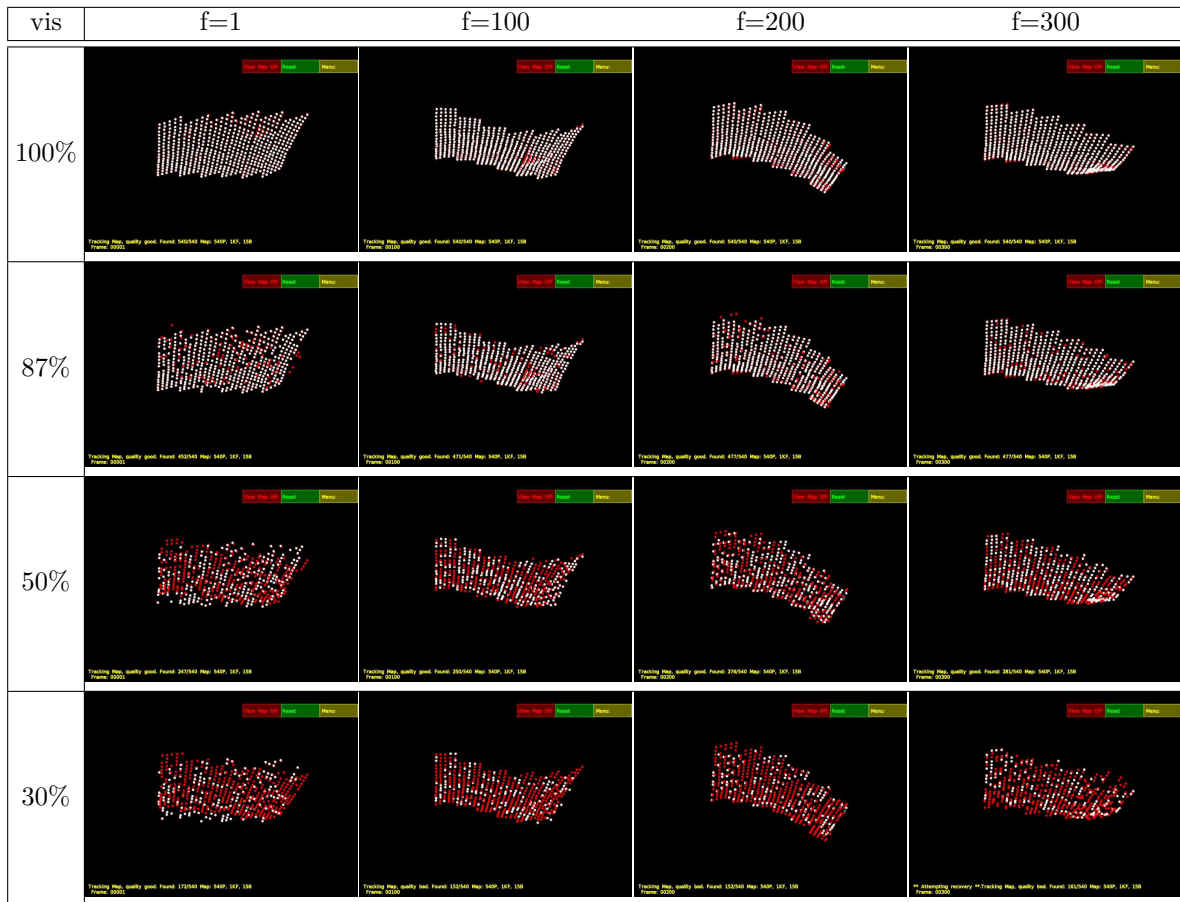


Figure 3.16: Flag sequence screenshots for visibility experiments

Some results within a range from 5% to 100% of visibility are shown in Fig. 3.17. As in the previous section, the four set of errors are presented, i.e. reprojection, reconstruction, rotation and translation. As the amount of points in this sequence is great compared to the CMUface, the error values are maintained stable from 10%, which means around 54 points, which appear to be enough to model the sequence with randomly distributed visibility masks and 15 bases. It has to be also noted that the visibility is modeled

randomly instead of being fixed in an area for a certain time, which could also affect the experiment.

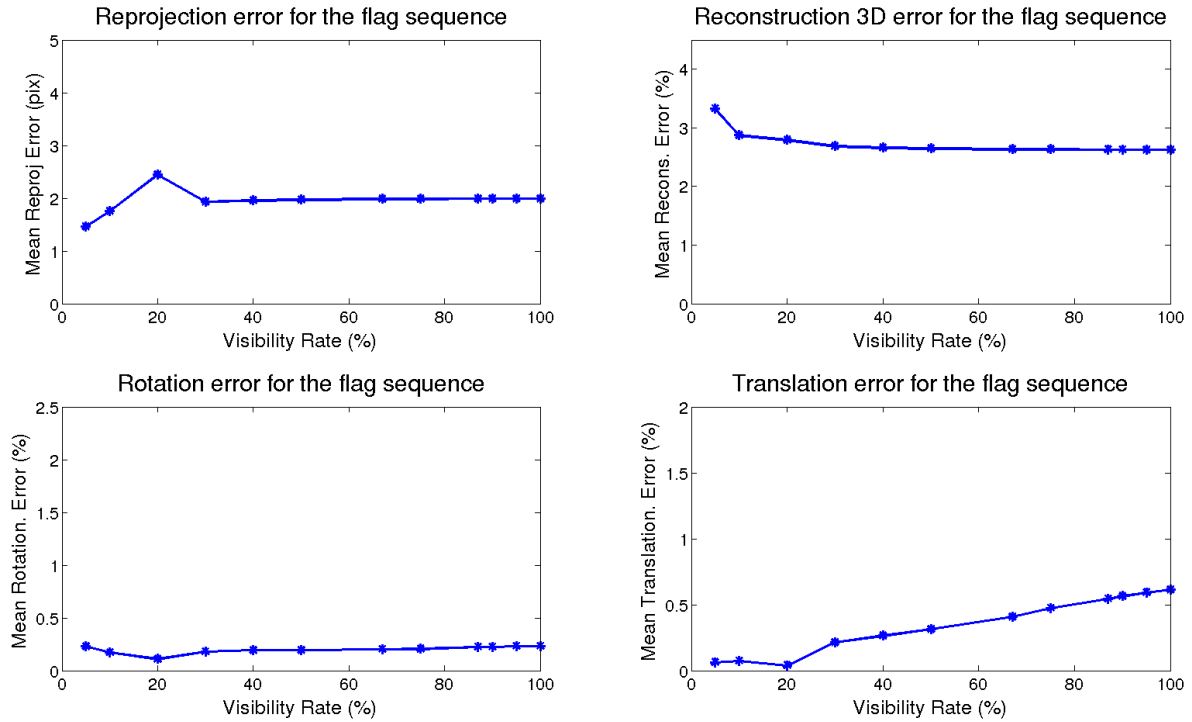


Figure 3.17: Flag sequence results for visibility degradation on tracking. Left-top subfigure depicts the variation with the percentage of visible points of the reprojection error. Top-right the same with the 3D reconstruction error. Bottom-left shows the dependence of the rotation error and in the bottom-right with the translation error.

### 3.2.2.2.3 Performance evaluation based on noise and outliers

Similarly to the previous sequence test, robustness analysis due to outliers and noise is performed to get closer to real condition. Some screenshots of the experiments are shown in Fig. 3.18.

It can be seen that, even though the measurements used in the tracking algorithm are not so good, the approximate reconstructed shape is accordingly approximated. These screenshots are shown to have a first idea about what degradation the system can afford.

Different values for noise and outliers have been evaluated, as commented in section 3.2.1. A study for these parameters is carried out, repeating the experiments 10 times and taking the average values. Their results are shown in Fig. 3.19. It can be seen in both Figs. (3.18 and 3.19), that the reprojection error is not incremented so much, and the gap is approximately preserved among approaches. With respect to the 3D reconstruction error, for  $\sigma = 0$  to 2 start about 2.6% and then all are growing up. The

3D reconstruction error shows a compensation of the effects of noise on 2D tracks for low noise levels. Unfortunately, it does not happen with outliers.

The rest of the error figures (rotation and translation) are stable within a range of 0.2-0.8% for rotation and 0.15-1.1% for translation, appearing the traces mixed up.

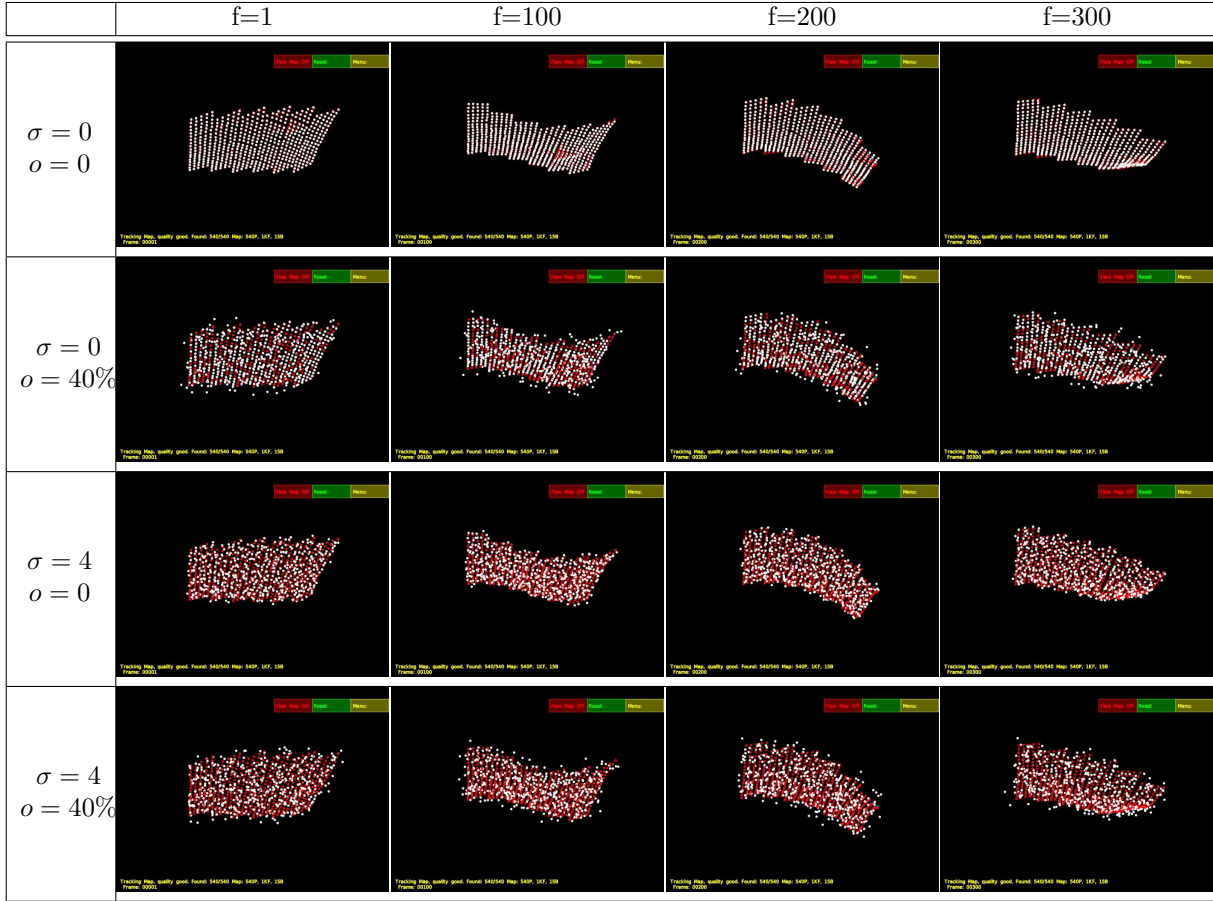


Figure 3.18: Flag sequence screenshots for noise and outlier experiments

#### 3.2.2.2.4 Performance evaluation based on the number of bases

As previously was done in the CMUface sequence, the dependence of the accuracy results with the number of bases is checked for the Flag sequence. For that purpose, a test using tracks without degradation and with 7, 15 and 30 bases is deployed.

With this setup, the results are shown in Table 3.3.

As expected, with perfect tracks, as the number of bases increase the error decrease.

To confirm the results depicted in the Table 3.3, the Fig. 3.20 illustrates the analysis over time.

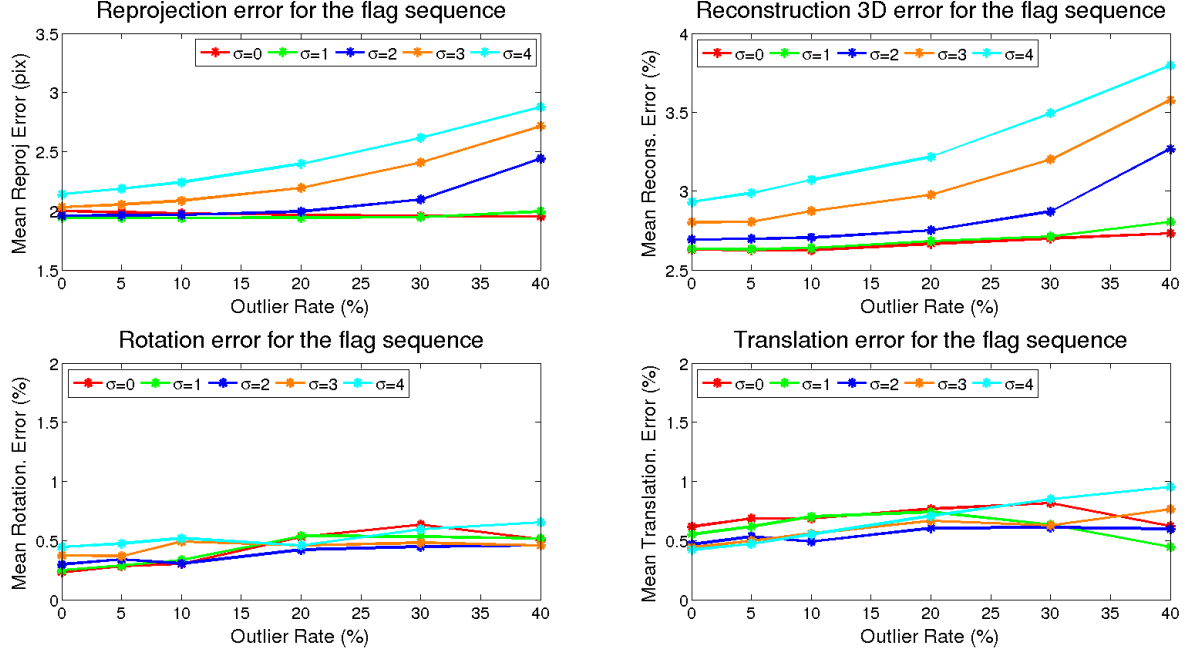


Figure 3.19: Flag sequence results for noise and outliers degradation on tracking. Left top subfigure depicts the variation with the  $\sigma$  percentage of outlier points of the reprojection error, each trace corresponds to a different noise  $\sigma$  level. Top right the same with the 3D reconstruction error. Bottom left shows the dependence of the rotation error and in the bottom right with the translation error.

#Bases( $K$ )	2D error (pix)	3D error(%)	rot. error(%)	trans. error(%)
7	3.62	3.21	0.53	1.89
15	2	2.63	0.23	0.61
30	1.18	1.93	0.07	0.23

Table 3.3: Flag sequence error comparison with the number of bases, for 7, 15 and 30.

### 3.2.2.2.5 Comparison with other methods of the state-of-the-art

Table 3.4 shows a summary of results among most representative methods of the state-of-the-art with sparse models. Processing time for the whole sequence, 3D accuracy, 2D error, rank, modelling type (auto / priori) and procedure type (batch / sequential) are depicted. Even though our proposal does not get the best 3D accuracy, neither the best reprojection error for this sequence (although they are very close to the best), it gets the best computation times, reaching the best trade off between performance and processing time.

Algorithms where the processing times are not available like [Vicente and Agapito, 2012], [Russell et al., 2014], etc., is due to processing time is not priority in their analysis. Batch algorithms implemented in Matlab like [Paladini et al., 2009, Paladini et al., 2010,

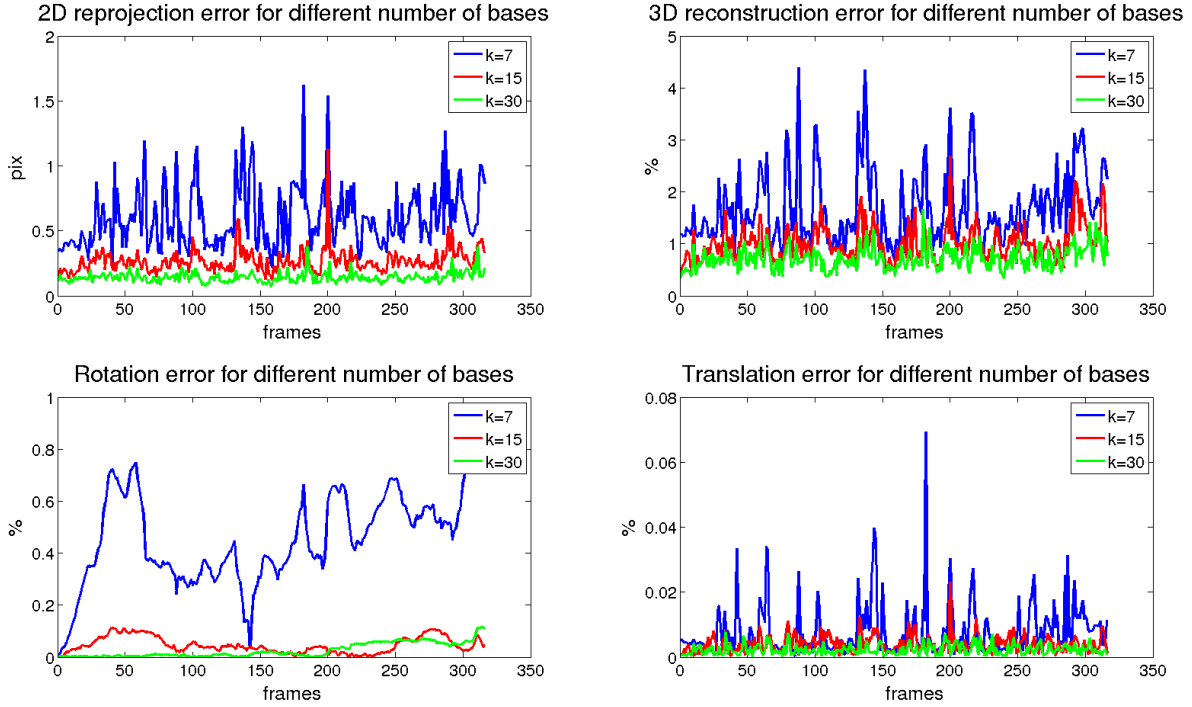


Figure 3.20: Flag sequence error over time varying the number of bases. The four error figures are shown. Top left shows the 2D reprojection error. The top-right shows the 3D reconstruction error. The bottom left shows the rotation error and the bottom right shows the translation error

[Torresani et al., 2008, Dai et al., 2012, Lee et al., 2016, Gotardo and Martinez, 2011] take more time than those developed in c++ language, like [Agudo et al., 2016a, Agudo et al., 2014].

Looking at the processing times in Table 3.4, it gives an idea of the data set complexity, even if most of the approaches are implemented in Matlab. This table shows that current solutions based on model-free methods (NRSfM) cannot recover a fully correct 3D shape for the flag sequence with a non-normalized error, unless the approach is piecewise (we recommended the readers a review of [Lee et al., 2016]) as the problem is highly ill-posed, and its manifold dimensionality very high. As our method is model-based, it is quite faster than model-free approaches, reaching 90 fps, which is compatible with the real-time constraint. It must be noted this time is reached without visual information processing, as in the rest of evaluated proposals.

The main problem of [Paladini et al., 2010] is that tries to model the object with a 1-rank shape for each frame, so 3 bases are needed for each frame. There is a need of a frequent re-factorization of the bases, which finally leads to over-fitting, causing high 3D error reconstruction due to the model fits the noise.

The trials of [Dai et al., 2012, Lee et al., 2016] did not finish to process the sequence after the referred time. For the methods where the source code is available [Paladini et al., 2009, Paladini et al., 2010, Gotardo and Martinez, 2011, Lee et al., 2016, Dai et al., 2012, Torresani et al., 2008], the algorithm is run, the processing time is measured in the

same hardware conditions, and the rank, if available is indicated, as well as the obtained error parameters. For the rest of the methods the results are extracted from [Lee et al., 2016] paper (check footnote 1<sup>1</sup>).

Method	2D error (px)	3D error (%)	rank	tproc / fps	model	proc. type
[Paladini et al., 2010]	6.14	91.65	7 (max)	20 min / 0.37	auto	seq
[Gotardo and Martinez, 2011]	29.79	66.65 / 16.09 <sup>1</sup>	-	36 min / 0.21	auto	batch
[Torresani et al., 2008]	11.65	15.59 / 13.25 <sup>1</sup>	4	10 h / 0.01	auto	batch
[Paladini et al., 2009]	<b>1.9</b>	29.724 / 17.61 <sup>1</sup>	9	19 seg / 23.68	auto	batch
[Vicente and Agapito, 2012]	-	<b>1.79</b>	-	-	auto	batch
[Russell et al., 2011]	-	<b>1.29</b> / 3.25	<sup>1</sup>	-	auto	batch
[Lee et al., 2016]	~ <b>2</b>	3.8	-	»28h	auto	batch
[Dai et al., 2012]	-	17.41 <sup>1</sup>	-	»16h	auto	batch
[Agudo et al., 2014]	-	3.28(10) 2.81(40)	10-40	19.5 init 1.53(10)f seg 2.28(40)f	auto	seq
Our approach	<b>2</b>	<b>2.63</b>	15	<b>5 seg / 90</b>	priory	seq

Table 3.4: Flag sequence results summary. Some results comparing several state-of-the-art algorithms and the presented one is depicted

### 3.2.2.3 Rendered Flag sequence

The flag sequence depicted in the previous tests is interpolated to get a dense sequence and then is rendered using orthographic projection, as presented in [Garg et al., 2011]. As our model-based tracking works on perspective projection, this sequence was re-rendered with perspective projection. This sequence is reconstructed for all the 450 original frames, instead of the 60 first presented in [Bronte et al., 2014]

This sequence was chosen in other works [Garg et al., 2011, Paladini, 2011, Garg et al., 2013, Agudo et al., 2014, Bronte et al., 2014] as it consists of a set of images and a dense 3D ground truth to evaluate the performance of tracking methods. It is useful for our experiments because, even though our tracking is sparse, the ground truth is appropriate to perform a thorough comparison among different methods.

In contrast to the guideline followed in the previous experiments, as there are several tracking proposals involved (PTAM based and descriptor based), the organization of the point will be different.

First of all, some frames of the sequences without processing are shown in Fig. 3.21 together with screenshots of the tracking operation using the different tracking proposals. The detected points are shown in magenta, the matched points in red and the mesh model is overlaid in cyan, to give an idea of the the tracking performance.

<sup>1</sup>In [Lee et al., 2016], the authors report other values of 3D reconstruction methods [Gotardo and Martinez, 2011, Torresani et al., 2008, Paladini et al., 2009, Dai et al., 2012]. They claim they tuned the setup to be optimal, but they neither specify how long do the experiments took nor the setup of their trials nor the configuration parameters



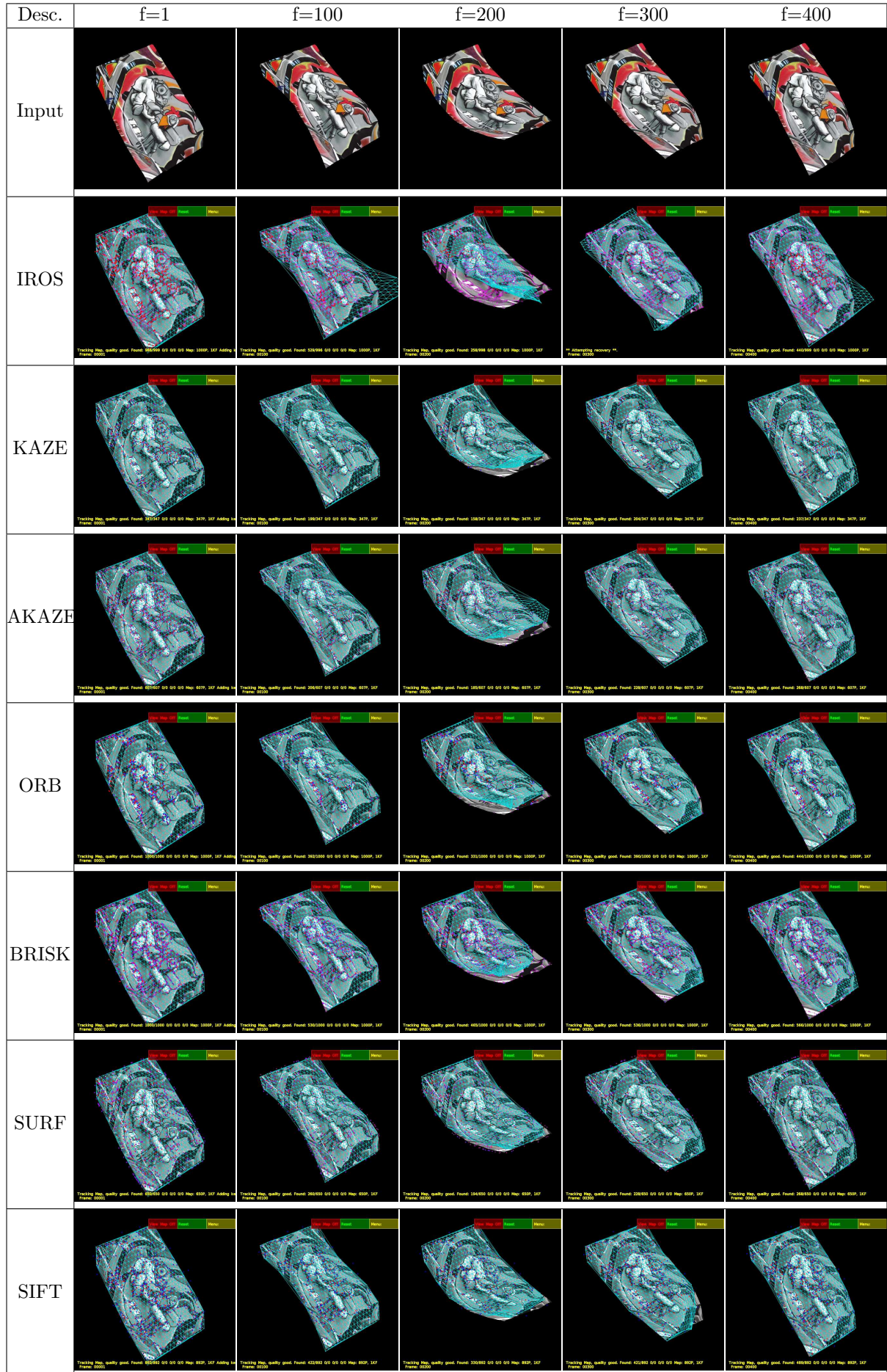


Figure 3.21: Rendered flag sequence screenshots



### 3.2.2.3.1 Visual descriptor evaluation

Except the proposal labeled as *IROS*, all the rest screenshots, related to feature descriptors, are displayed with the *Bruteforce* matching included in OpenCV. On the benchmark shown in Fig. 3.23, the rest of the matching variants implemented in OpenCV are used (L1, L2, Hamming distance).

The frames are organized on columns, whereas the descriptor types are on rows. The *IROS* uses the PTAM-based pyramidal descriptor proposal modified by the authors, as was presented in [Bronte et al., 2014]. The rest of rows shows the type of feature descriptor used in the detection step. Depending on the type of descriptor the distribution of the points are more or less homogeneous, stable, repetitive, etc, because it depends on the characteristics of each of the features.

The *IROS* is prone to lose a track when the movement on certain points is higher than 30 pixels (which happens on the lower parts of the waving flag). The matching position is not reliable for points very far from the previous frame points, whereas feature based matching is more reliable. This can be easily seen in the frame #200 of the Fig. 3.21, when comparing the *IROS* with the rest of approaches. The model representation (in cyan) is much adjusted in the feature based proposals than in the *IROS* proposal.

Depending on the selected feature, some areas are not properly covered whereas other features are more uniformly distributed on the image. The features are configured to be as fast as possible, which could lead to a penalty in the quality of points detection, because we are looking for a compromise between quality and execution time.

Reconstruction results are provided in Fig. 3.22. For each of the frames two views of the 3D reconstruction are given. Similarly to the previous figure, in the columns are given the frames and in the rows the descriptors. It can be seen that in frame #200 and #300 there are some accuracy problems for most of the proposed descriptor. This is due to the sequence presents very abrupt movements combined with an interval of possible tracking losses.

The more uniformly the area is covered, the best the reconstruction will be, therefore better distributed descriptors are preferred against the ones that are very concentrated in an area. This is one of the weak points of some feature-based tracking algorithms.

The projection of the model on the screenshots figure (Fig. 3.21) in cyan and the reconstruction results shown in Fig. 3.22 can vary, because the shape in the latest figure is obtained applying Procrustes alignment between the ground truth and the reconstruction. Even with this alignment, there are some intrinsic errors due to the NRSfM / SfT problem is intrinsically ill-posed when no-priors are applied to better conform the problem.

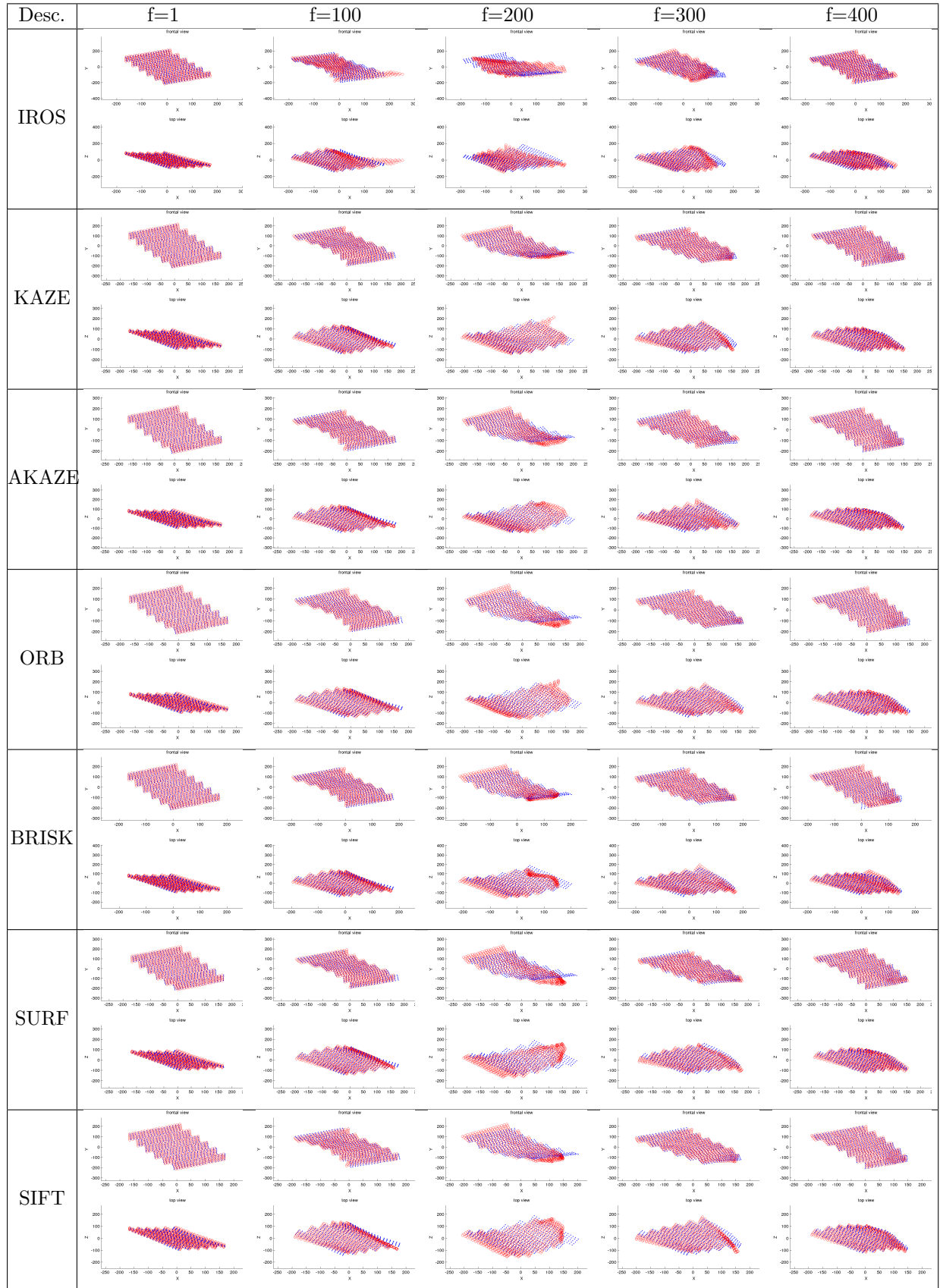


Figure 3.22: Rendered flag sequence reconstruction results compared with ground truth for the different descriptors

In order to have a general performance overview for all the possible descriptors, a benchmark is tackled in the Table 3.5. This comparative includes number of points, the descriptor the matching technique, processing time (for the whole sequence and fps), 2D error, 3D error, rotation error and translation error.

Descriptor	Matcher	2D error(px)	3D error(%)	rot error(%)	t. error(%)	tp (s/fps)	map Pts
PTAM	PTAM	40.12	104.54	54.51	99.97	102 / 4.4	350
IROS	PTAM-like	9.51	16.65	4.24	2.56	<b>26 / 17.3</b>	972
AKAZE	BruteForce	5.68	12.42	<b>2.86</b>	2.41	<b>27 / 16.6</b>	607
BRISK	BruteForce	5.98	14.27	3.45	2.47	56 / 8	1000
ORB	BruteForce	5.67	12.71	4.0	2.52	29 / 15.5	1000
KAZE	BruteForce	5.55	13.12	4.01	<b>2.37</b>	49 / 9.2	347
SIFT	BruteForce	<b>5.27</b>	<b>12.04</b>	3.80	2.39	87 / 5.2	892
SURF	BruteForce	5.4	14.11	3.91	2.6	29 / 15.5	650

Table 3.5: Rendered flag sequence descriptor results comparative. As a references, PTAM original algorithm, the PTAM-based tracking and the descriptor based results are presented.

A similar benchmark but including the type of matching used for each of the feature descriptors is shown in Fig. 3.23. This studies the influence of the matching method on the error metrics. For binary descriptors such as AKAZE, ORB, and BRISK the use of Hamming distance slightly improves the results of 2D reprojection and 3D reconstruction, errors having no effect or getting worse results for rotation and translation errors. Using the rest of the metrics, results are similar for all the errors except some few cases.

It can be seen on Fig. 3.23 that the best method for the reprojection error is not necessary the best for the reconstruction error, as we said before. SIFT gets the best marks for both error estimations and matching algorithms, although the second best is not the same in both figures. For the other two errors is among the best.

AKAZE performs almost as well as SURF in 2D reprojection error, is almost as good as SIFT in 3D reconstruction error, and for the rest of the errors gets also good marks. Looking at the Table 3.5 it can be seen that is one of the fastest approaches, so it is a good candidate for a final robust implementation.

Regarding this table, it can be seen that the fastest approaches are *IROS*, AKAZE, ORB and SURF and the ones that handle the most points are *IROS*, BRISK, ORB and SIFT. Remember that state-of-the-art approaches are only capable of working with a very little amount of points obtained an from interpolated model. In this case all the detected points are used to perform the estimation. It is important to remark that most of the reported time is consumed by the feature detection algorithm. The timing show the influence of the visual processing, comparing the reported results for the Flag sequence in Table 3.4 for the case where the visual detection stage is not applied.

The sequence trajectory reconstruction is depicted in Fig. 3.24. It is rendered with the camera fixed, but the dense ground truth comes with a little movement, which is

shown in Fig. 3.24, which confirms that the estimation of the pose works jointly with the deformation estimation. Due to this camera pose cannot be directly evaluated, we consider the camera reference static and compare results against this static reference.

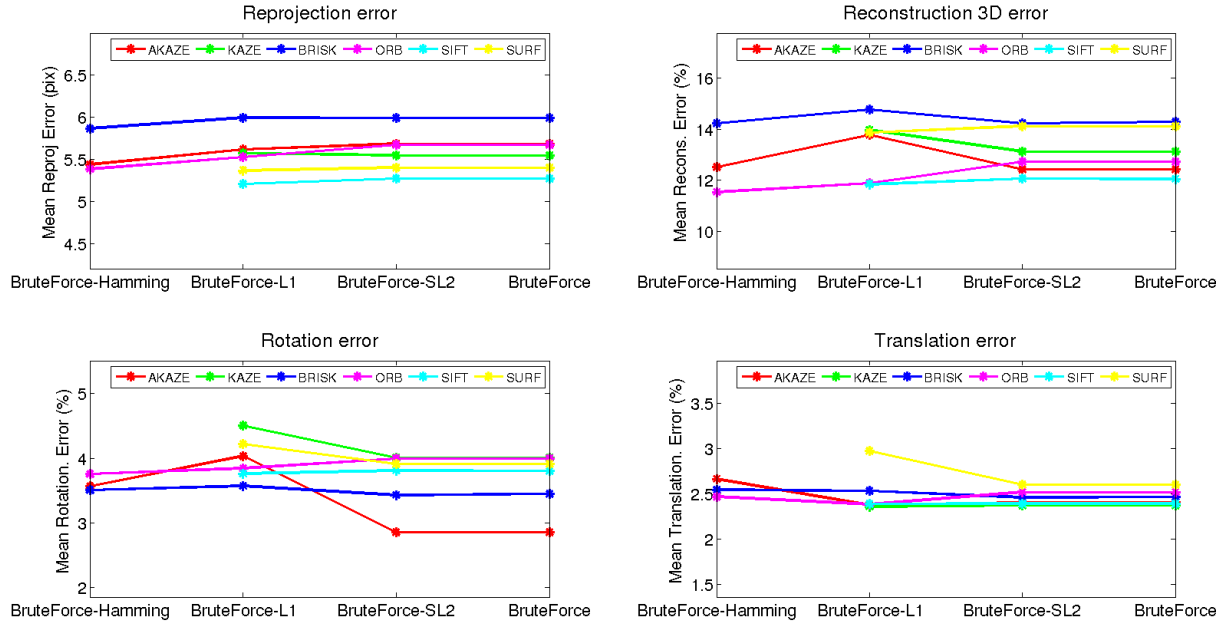


Figure 3.23: Rendered flag sequence rank based on the error results of the different descriptor methods. Top left reprojection error, top right 3D reconstruction error, bottom left rotation error, bottom left translation error.

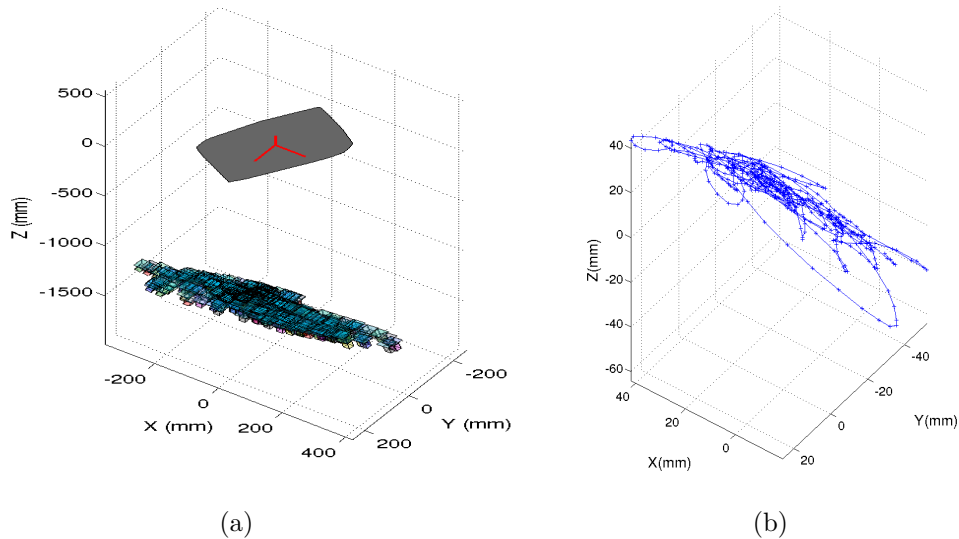


Figure 3.24: Trajectory estimation for Rendered flag sequence. SIFT descriptor example. (a) the reconstructed trajectory. (b) The average displacement extracted from the ground truth of the sequence, i.e. an intrinsic translation which evidences a relative movement between the camera and the object position.

### 3.2.2.3.2 Performance evaluation based on time and shape priors

Working with real images increase uncertainty in the estimations. In order to reduce the effects of ambiguities on the estimation, get more accurate 3D estimations, avoid flickering due to erroneous estimations, etc, priors must be added. Due to the nature of our optimization proposal (linear and split), time and shape smoothness can be implemented.

To improve results, the priors are included, in 3 levels (low, medium and high). Low means the effect of the prior is almost unnoticeable, using the medium value, the effect is not so high, and a high value makes the shape rigid for both priors.

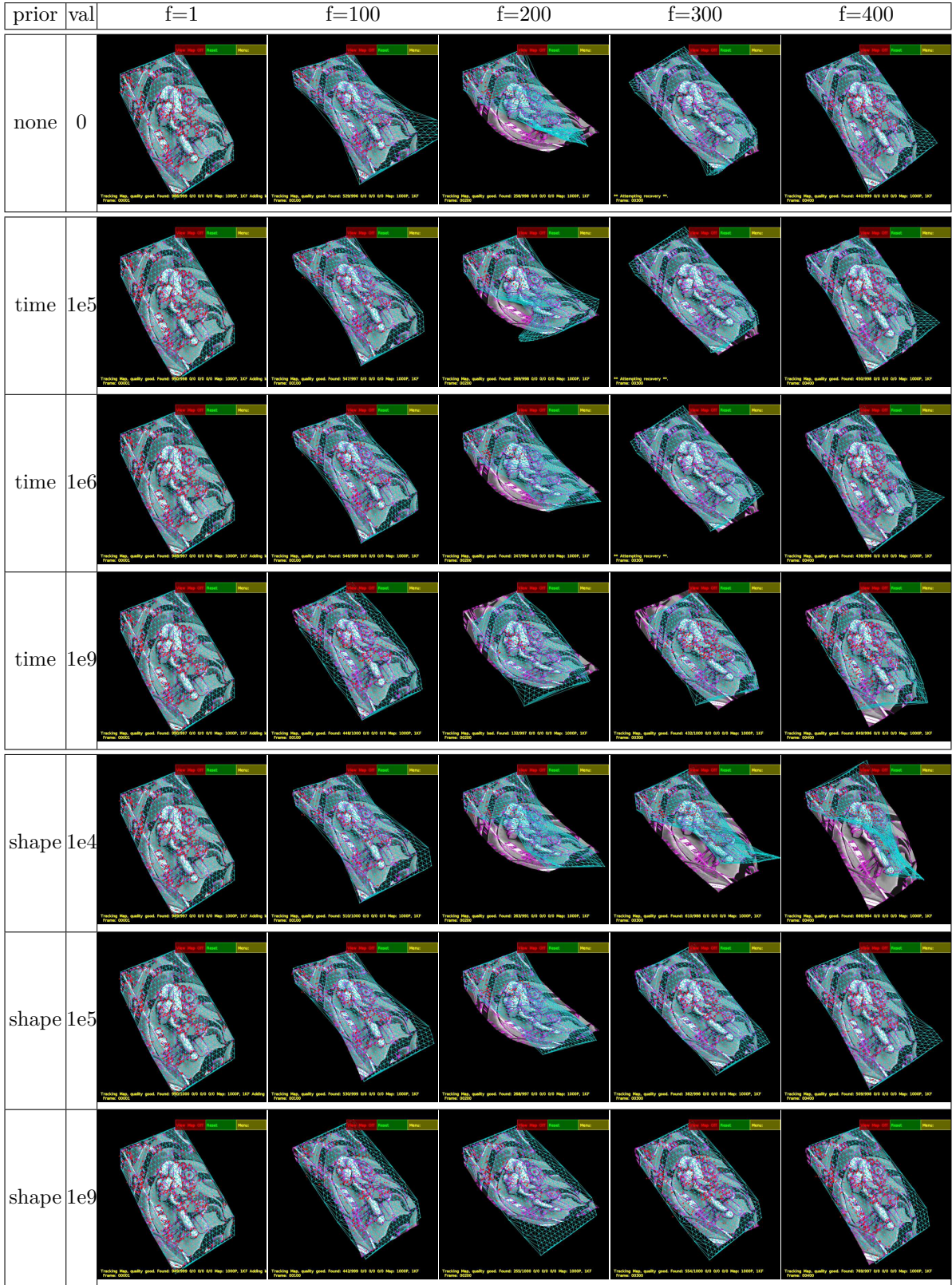
Only the SIFT descriptor is chosen, as it is a reference on the state-of-the-art has obtained the best performance results in our previous tests, and its working is similar for the rest of the descriptors. Then, the results are compared against the *IROS* approach because this use other tracking strategy and serves as a base, bring our first implementation [Bronte et al., 2014].

Snapshots for different priors are shown in Fig. 3.25 for the *IROS* tracking, and in Fig. 3.26 for the SIFT feature-based tracking. The corresponding reconstruction results are shown in Fig. 3.27 and Fig 3.28 respectively. As in previous figures, the frames are distributed in columns whereas the priors are specified in rows. The first row represents the experiment with no prior. The 3 rows below show the tests for time smoothness priors (low, medium ,high), and the last 3 rows the tests for the shape smoothness prior (low, medium, high).

The behavior for the highest priors for both tracking approaches are similar, and can be checked on frame #200. The prior makes the shape rigid over time so the pose is the only parameter that is adjusted in the sequence. A more detailed study will be addressed on the temporal analysis of the sequence.

Some of the beneficial effects of priors can be seen in different frames. For the frame #100, compared with the first row, priors have the effect of reducing dispersion in the reconstructed corners of the flag. For higher deformations, as the presented in #200, it can be seen that dispersion is compensated, although not completely corrected. The effects are less noticeable for SIFT-based tracking than for *IROS* based tracking, because prior feature-based tracking quality is higher than for the *IROS* tracking, making that the improvement margin for the first case was lower.



Figure 3.25: Rendered flag sequence screenshots when priors are applied for *IROS*-based tracking

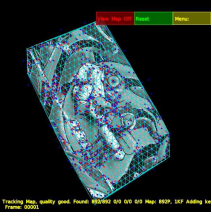
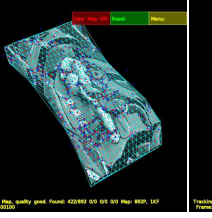
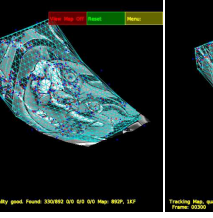
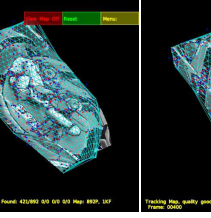
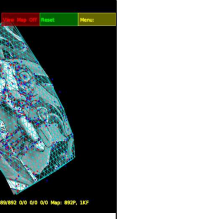
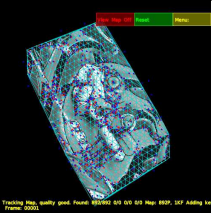
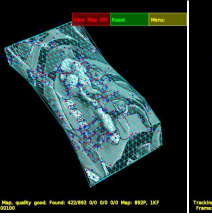
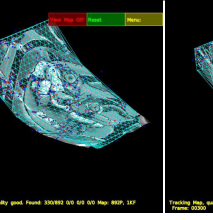
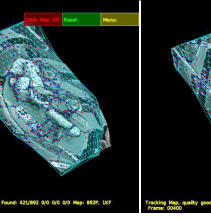
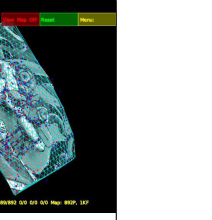
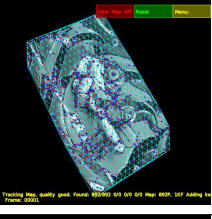
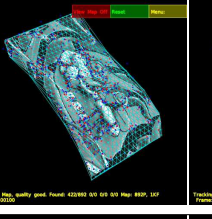
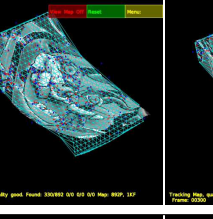
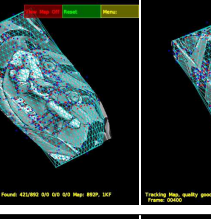
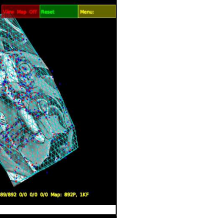
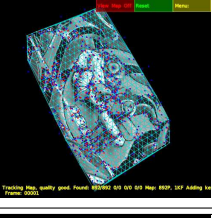
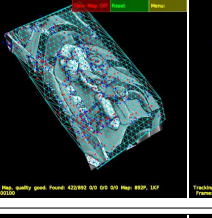
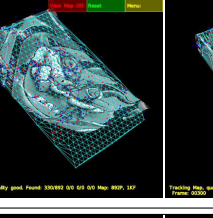
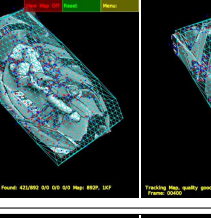
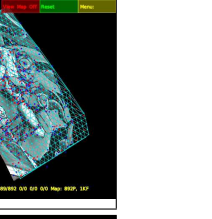
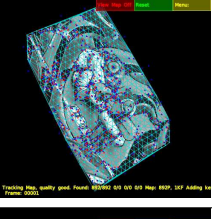
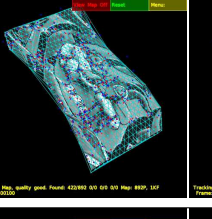
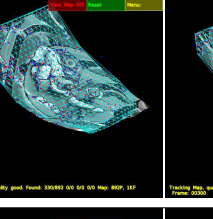
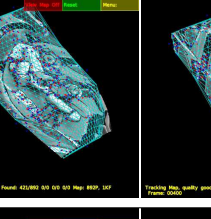
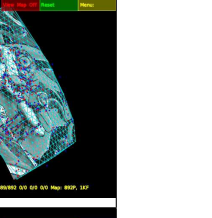
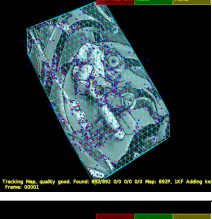
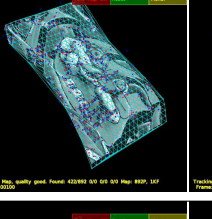
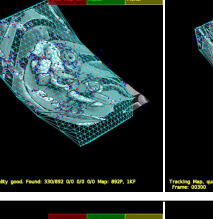
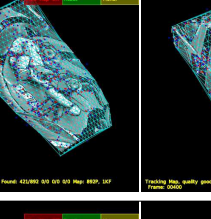
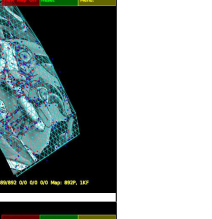
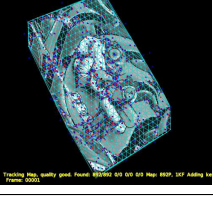
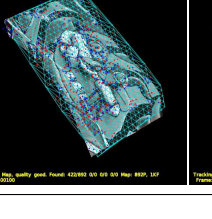
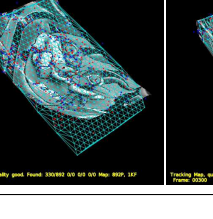
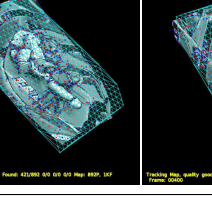
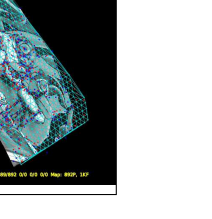
prior	val	f=1	f=100	f=200	f=300	f=400
none	0	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
time	1e5	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
time	1e6	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
time	1e9	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
shape	1e4	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
shape	1.5e5	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000
shape	1e9	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 330392 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 432892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000	 Tracking Map, quality good, Found: 488892 0.0 0.0 0.0 Map: 852P, 147 Frame: 0000

Figure 3.26: Rendered flag sequence screenshots when priors are applied for SIFT descriptor



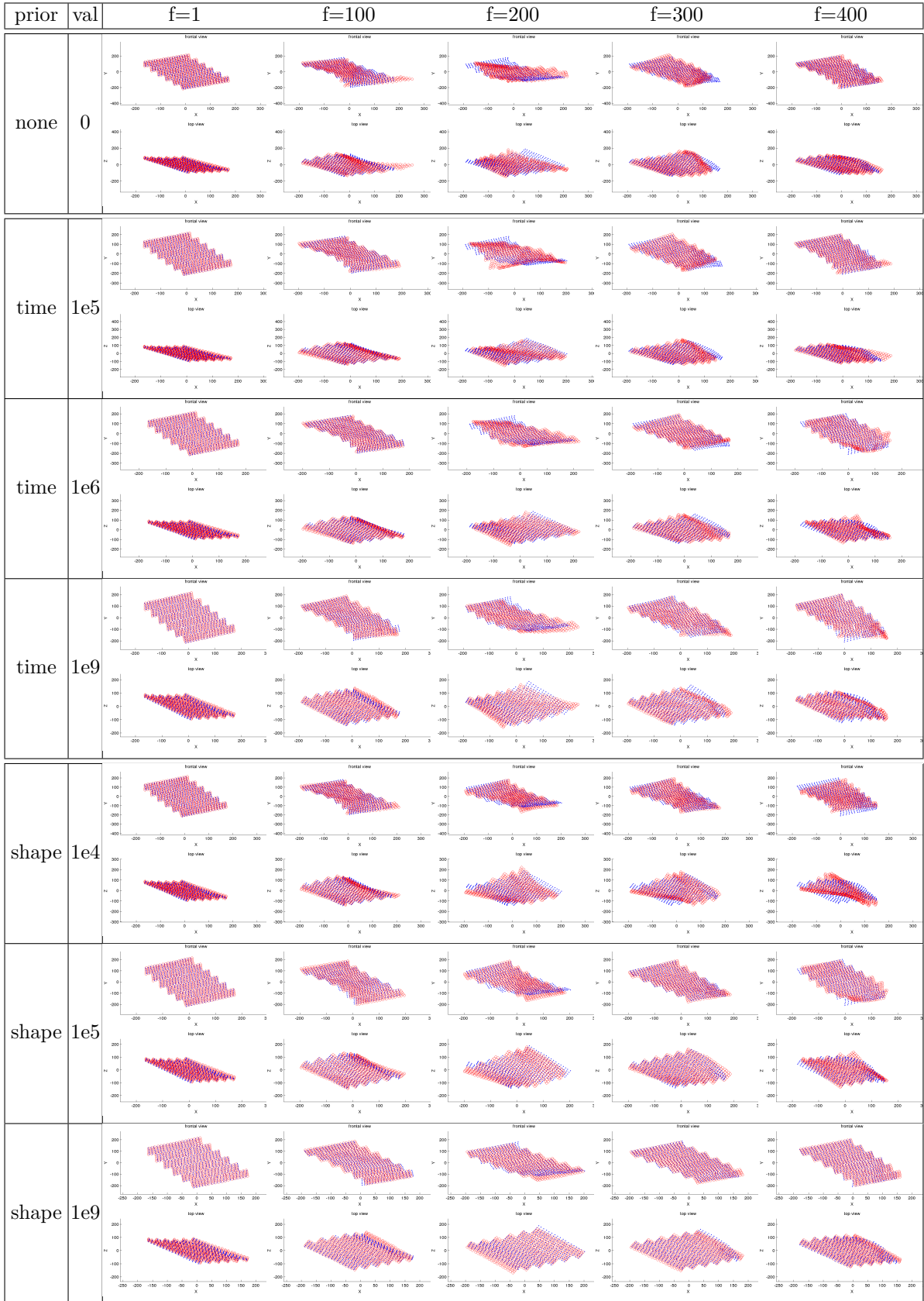


Figure 3.27: Rendered flag sequence reconstruction results compared with ground truth for the *IROS* descriptors



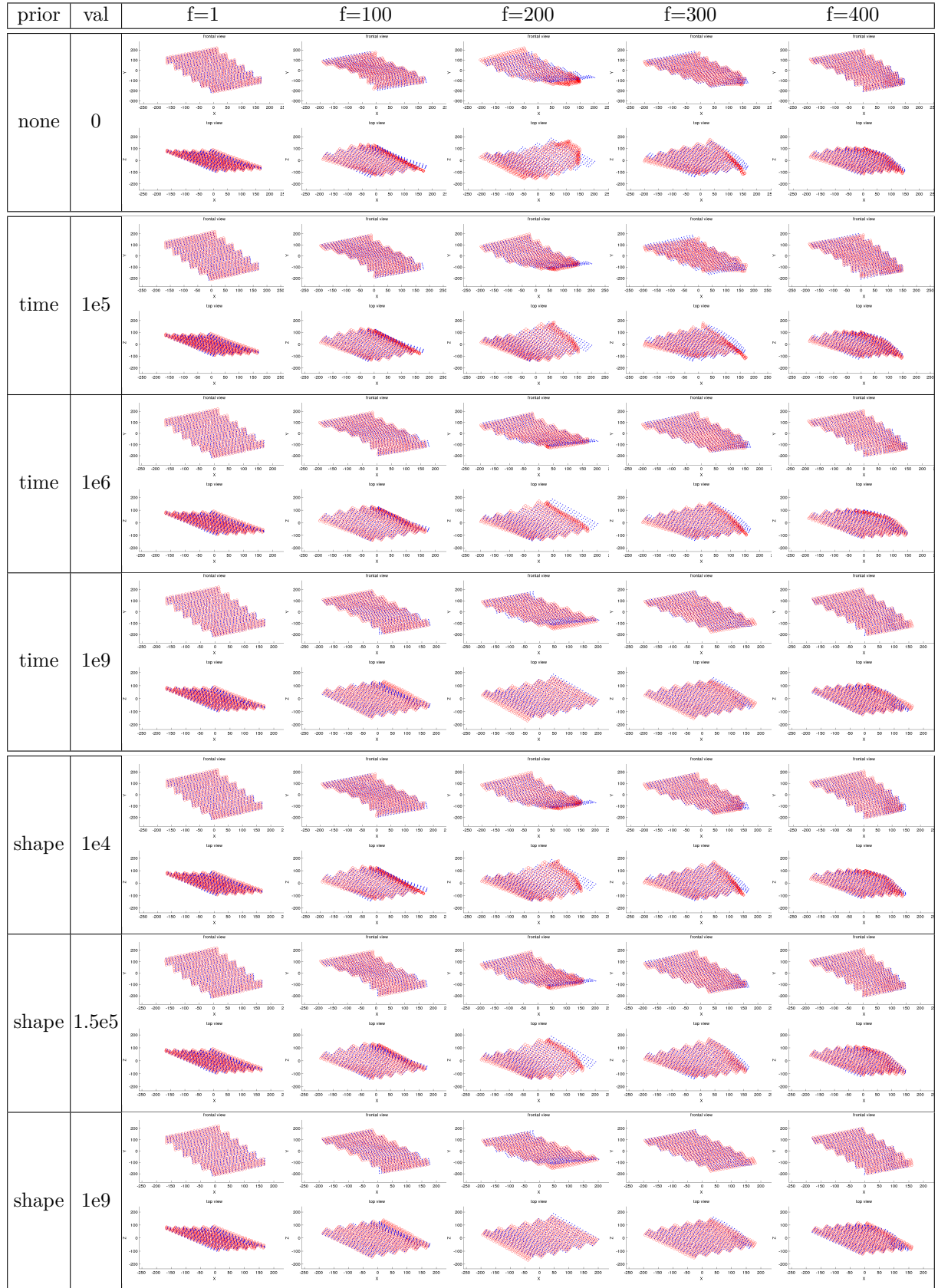


Figure 3.28: Rendered flag sequence reconstruction results compared with ground truth for the SIFT descriptors

After 3D reconstructions and 2D projections have been studied, a temporal analysis of the tracking for both prior regarding the baseline without priors is carried out.

The first evaluated prior is time smoothness depicting all the errors shown in Figs. 3.29, 3.30, 3.31 and 3.32, which correspond to reprojection, 3D reconstruction, rotation and translation errors. The figures are divided in 2 parts: The upper part corresponds to the *IROS* tracking whereas the lower part corresponds to the SIFT tracking.

In Fig. 3.29 the reprojection errors are presented for the different priors, each in one trace. The blue trace represents the result with no priors. In terms of error, the cyan and red traces are always close to the blue trace and the green one presents a high error for both approaches, except when the tracking is lost, as an excessive value of the prior makes the shape so rigid. The cyan trace, as expected, has little effect in the reprojection error, as the regulator value is low. The red trace has a slightly higher reprojection error although in general terms it is maintained along the sequence. Therefore, for the reconstruction error, the prior does not imply an improvement.

The beneficial effect of the prior is seen in the 3D reconstruction, in Fig. 3.30. For both approaches, when the prior is applied, even for little values, the 3D reconstruction error is reduced when the great peaks appear.

The best option is the red level for both approaches, because the increment in the reprojection error is little and the improvement in the 3D reconstruction error is great with respect to the blue trace (no priors applied). The effect is even more noticeable when the tracking is better comparing the *IROS* and SIFT approaches.

Having a look at rotation and translation errors shown in Figs. (3.31 and 3.32), it can be seen that the highest prior (green trace) is the one that most error gets in several time slots of the sequence. With respect to the rest of the approaches, the error follows a similar trend as we explained above.

The results of the temporal analysis for the shape smoothness prior are depicted in Fig. 3.33 (reprojection), Fig. 3.34 (3D reconstruction), Fig. 3.35 (rotation) and Fig. 3.36 (translation). A similar conclusion as the temporal smoothness prior can be extracted for a time analysis of the shape smoothness prior. The trace in red, meaning an intermediate value of the prior, reflects an equilibrium between reprojection and reconstruction errors, for both matching methods (*IROS* and SIFT-based).

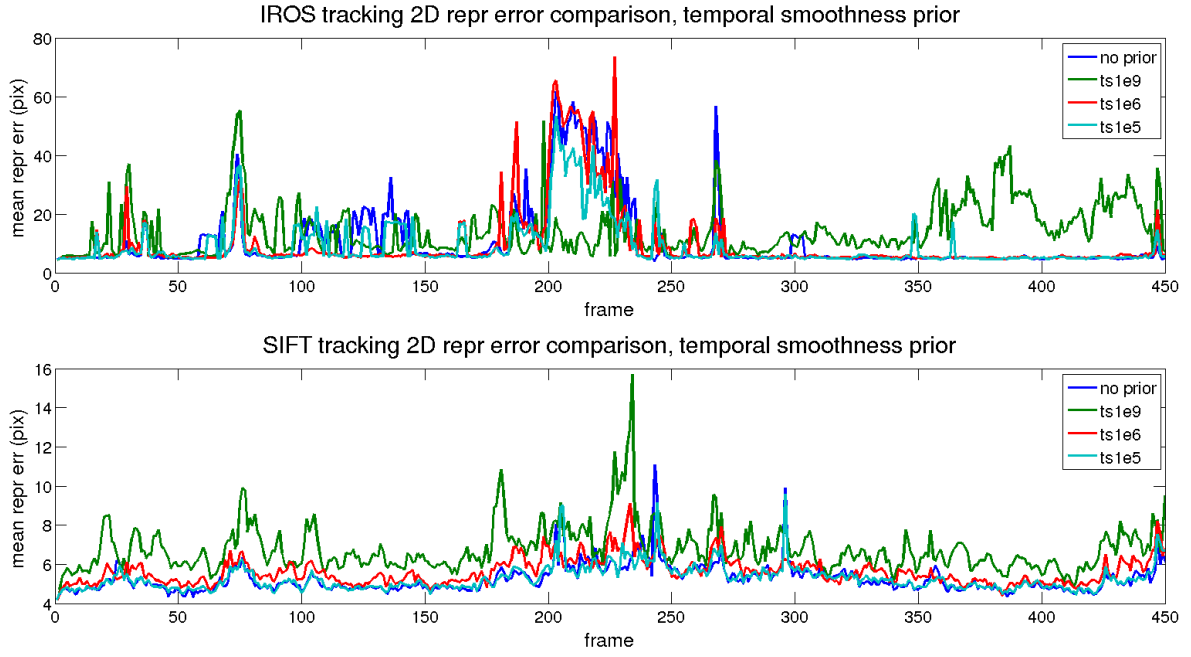


Figure 3.29: Rendered flag sequence **reprojection error** results over time when **time smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

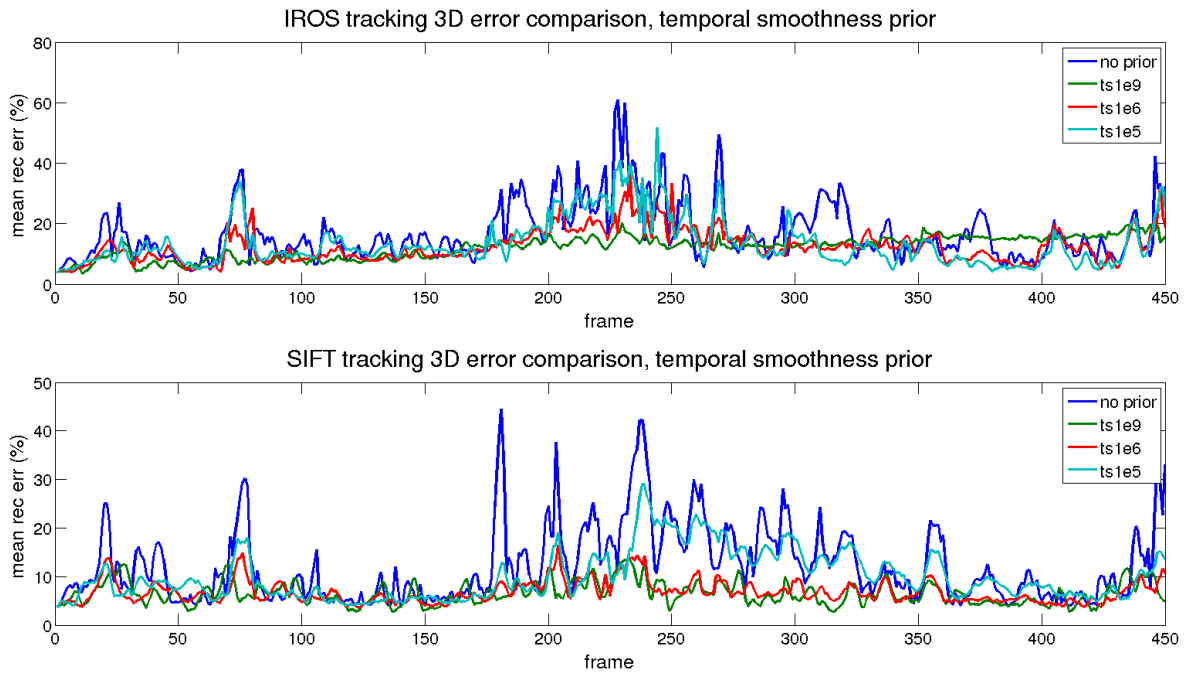


Figure 3.30: Rendered flag sequence **3D reconstruction error** results over time when **time smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

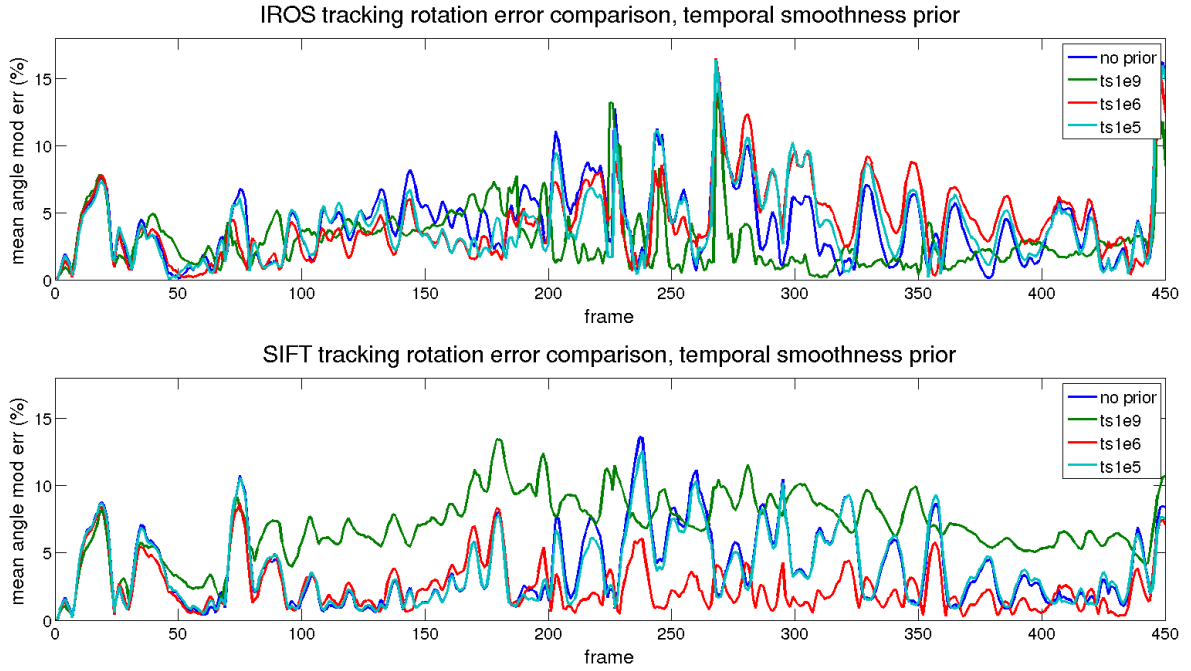


Figure 3.31: Rendered flag sequence **rotation error** results over time when **time smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

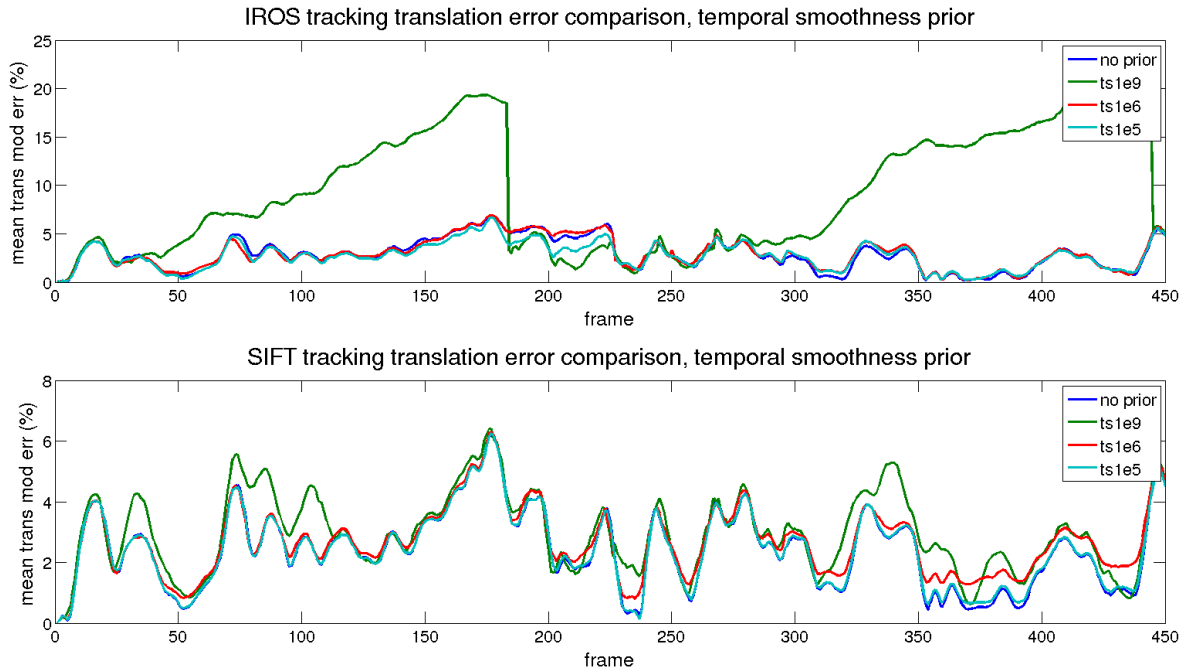


Figure 3.32: Rendered flag sequence **translation error** results over time when **time smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

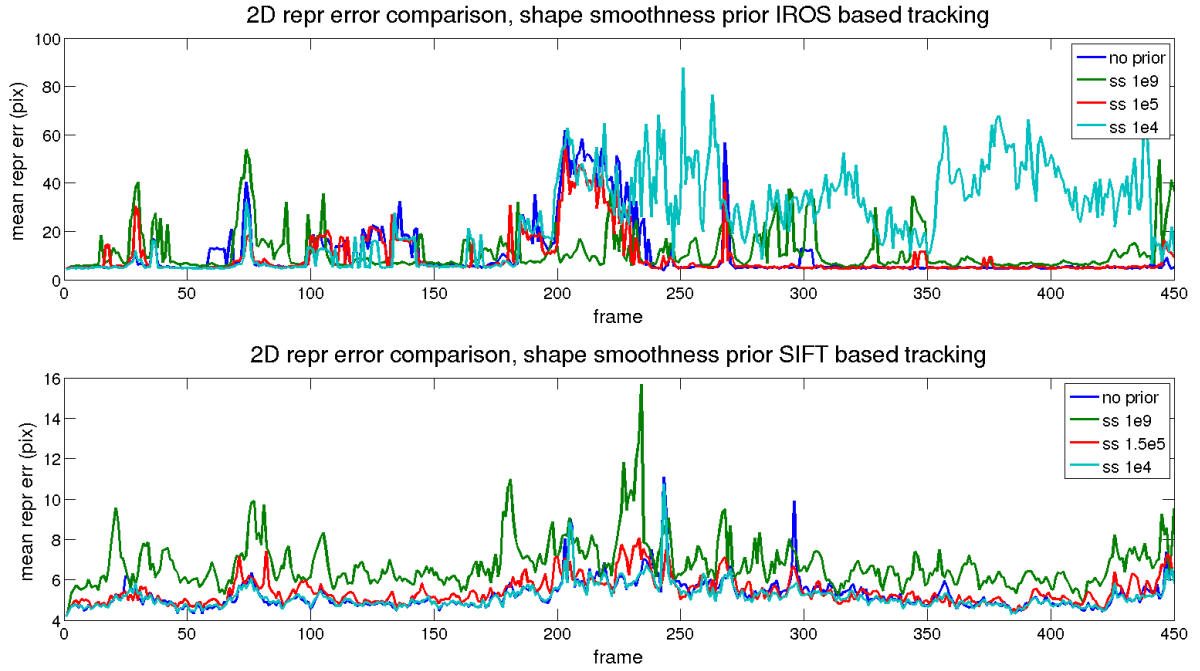


Figure 3.33: Rendered flag sequence **reprojection error** results over time when **shape smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

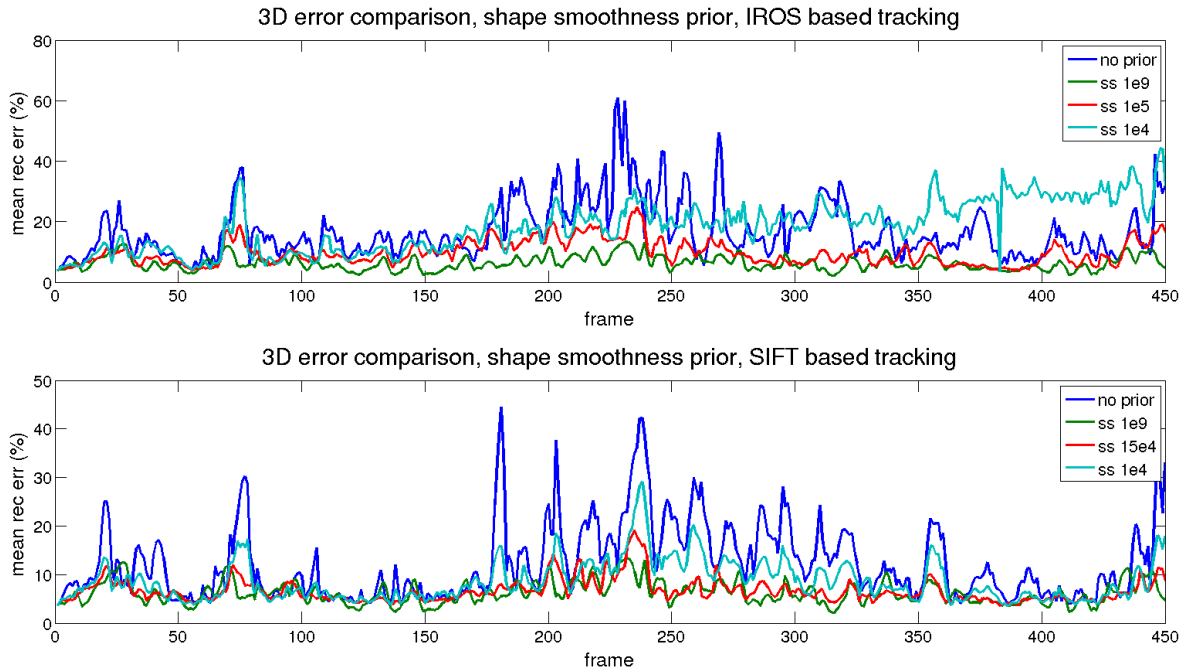


Figure 3.34: Rendered flag sequence temporal **3D reconstruction error** results when **shape smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

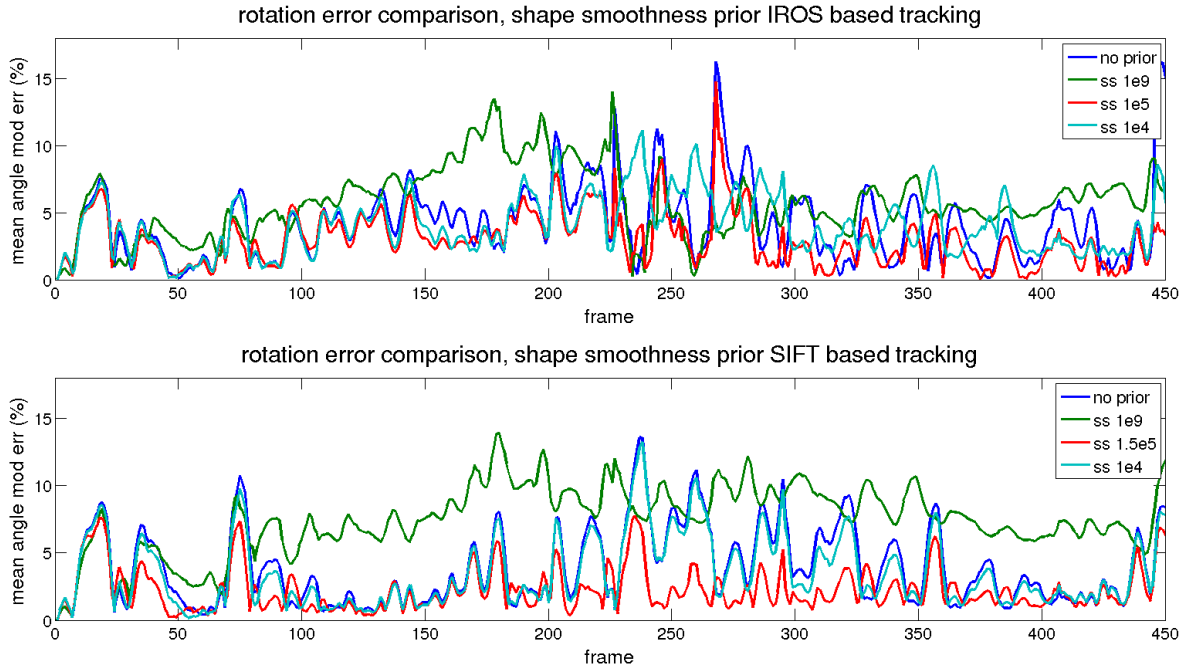


Figure 3.35: Rendered flag sequence **rotation error** results over time when **shape smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

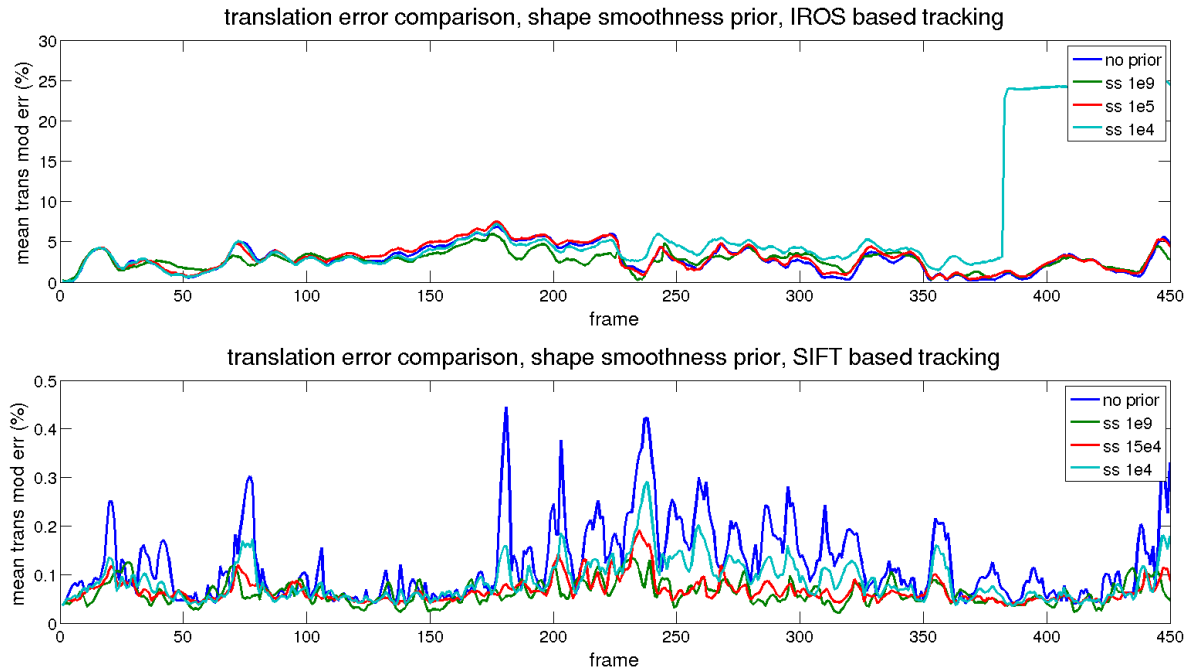


Figure 3.36: Rendered flag sequence **translation error** results over time when **shape smoothness** prior is applied for both descriptor types. From top to bottom: *IROS*, *SIFT*.

Trajectory examples for both tracking approaches and different priors are depicted in Fig. 3.37 regarding the no priors option. The priors level are represented in columns whereas in rows are represented the descriptor, prior type and the trajectory results. For the 'none' prior results are placed in the 'med' column to be centered in the figure.

It can be clearly seen that the highest priors affect the estimated trajectory on the depth and the dispersion in the other 2 axes. This is due to the prior is so high that the shape cannot deform itself, so the estimator tries to compensate the error only by variations of the pose, as any Structure from Motion (SfM) system would do in the same conditions.

On the *IROS* proposal, the trajectories for low priors do not improve a lot the estimation or even get it worse, as in the shape prior. For the SIFT proposal the trajectory estimation does not vary so much when the low and medium priors are applied.

To sum up, the errors reached after applying different priors are depicted in Table 3.6. The best results for each prior are marked in bold, which could not be the best for each column but is a compromise solution, among all the parameters involved on it.

Desc.	prior type	value	2D error(px)	3D error(%)	rot error(%)	t. error(%)
PTAM	none	0	40.12	104.54	54.51	99.97
Point-wise Perfect Matching	none	0	2	2.63	0.23	0.61
IROS	none	0	10.36	16.66	4.1	2.81
IROS	time	1e5	8.92	13.38	4.07	2.67
<b>IROS</b>	<b>time</b>	<b>1e6</b>	<b>9.76</b>	<b>12.42</b>	<b>4.23</b>	<b>2.90</b>
IROS	time	1e9	14.37	12.04	3.00	9.29
IROS	shape	1e4	24.05	18.31	3.98	6.62
<b>IROS</b>	<b>shape</b>	<b>1e5</b>	<b>9.38</b>	<b>9.54</b>	<b>2.96</b>	<b>3.01</b>
IROS	shape	1e9	11.62	6.2	5.76	2.66
<b>IROS</b>	<b>both</b>	<b>1e5/1e5</b>	<b>8.72</b>	<b>8.95</b>	<b>3.60</b>	<b>2.45</b>
SIFT	none	0	5.27	12.04	3.81	2.39
SIFT	time	1e5	5.29	9.93	3.71	2.41
<b>SIFT</b>	<b>time</b>	<b>1e6</b>	<b>5.64</b>	<b>6.79</b>	<b>2.48</b>	<b>2.66</b>
SIFT	time	1e9	6.79	6.51	6.95	2.95
SIFT	shape	1e4	5.18	8.48	3.45	2.41
<b>SIFT</b>	<b>shape</b>	<b>1.5e5</b>	<b>5.45</b>	<b>6.56</b>	<b>2.26</b>	<b>2.64</b>
SIFT	shape	1e9	6.80	6.2	7.44	2.86
<b>SIFT</b>	<b>both</b>	<b>1e5/1.5e5</b>	<b>5.48</b>	<b>6.46</b>	<b>2.05</b>	<b>2.57</b>

Table 3.6: Results summary for different prior types and the influence of the descriptor type used

PTAM results are included so to have a rigid method as reference that applies visual processing with rigid SfM. On the other hand, just on the row below, the results from the (point wise) Flag sequence with perfect matching (Table 3.4) are copied, to check how important is the influence of the visual processing on the results.

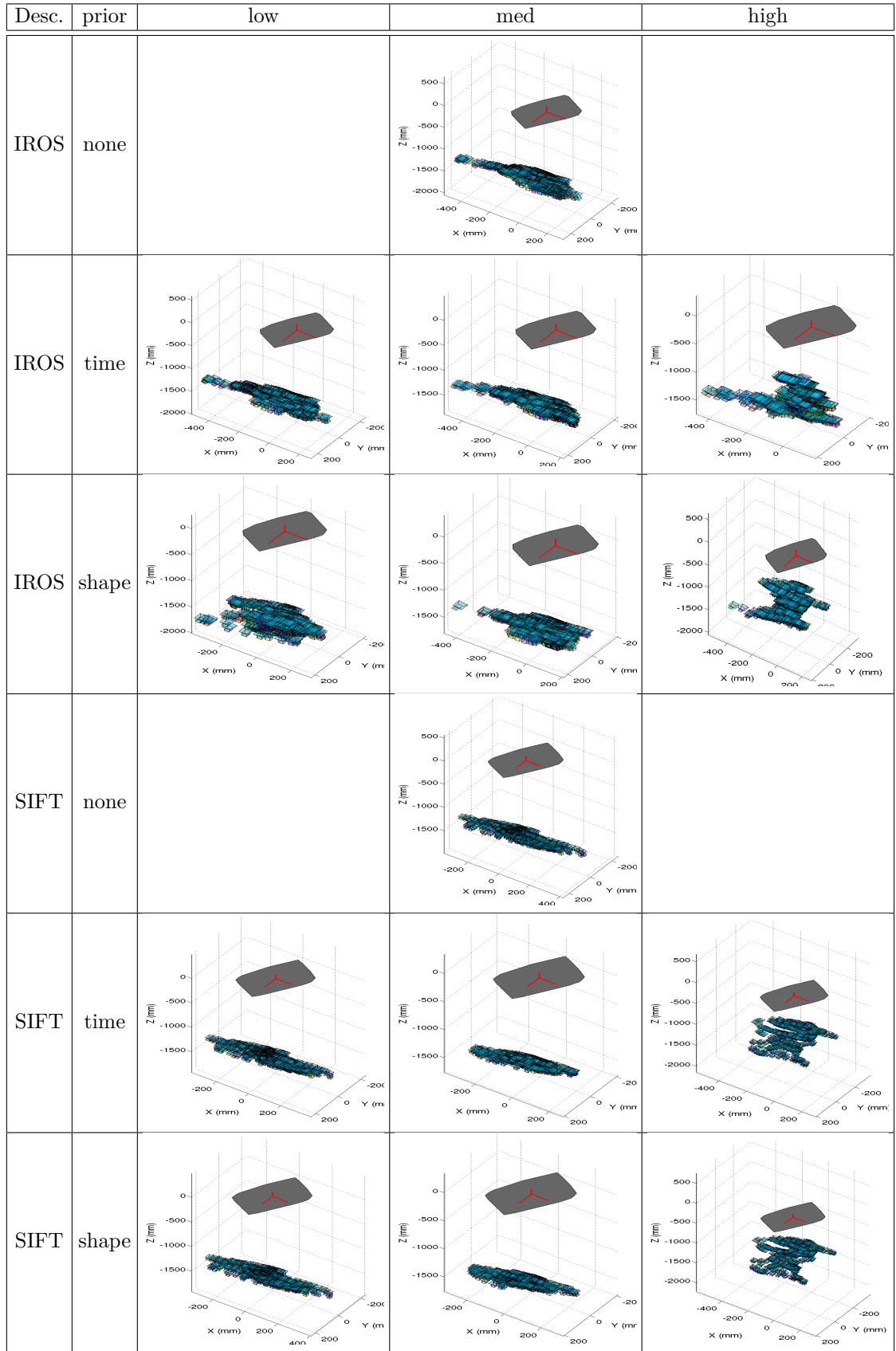


Figure 3.37: Trajectory representation for Rendered flag sequence for different time and shape priors



As far as the authors knowledge there are no SFT publications including results with this dataset.

The difference between *IROS* and SIFT approaches is easy to explain. Tracking quality is decisive to get accurate results, as the 2D reprojection error is about 5 pixels lower for SIFT before priors application, the 3D reconstruction error is about 4% lower (16 to 12) before priors and after priors it is reduced an additional 3% with respect to the best mark (9 to 6).

The processing time due to the priors is not included on the table, because it does not add a substantial change.

### 3.2.2.3.3 Performance evaluation based on the number of bases

As it is done previously with the other datasets, the influence of the number of bases with the estimation accuracy is tested for real conditions. To separate the evaluation of this parameter from other source of error, no priors are added.

First we show the results in Table 3.7. It can be seen the trend, unlike the previous sequences with ideal matching, is reversed, as with *IROS* based tracking the best results are obtained with 7 bases and the best 3D results are also obtained for SIFT for 7 bases.

Matching	#Bases	2D error (pix)	3D error (%)	rot err (%)	trans err (%)
<b>IROS</b>	<b>7</b>	<b>8.83</b>	<b>9.19</b>	<b>2.94</b>	<b>2.56</b>
IROS	15	9.81	16.91	4.99	2.57
IROS	30	13.2	34.23	5.77	2.61
SIFT	7	5.76	<b>6.53</b>	<b>2.65</b>	3
SIFT	15	5.6	12.26	3.91	2.52
SIFT	30	<b>5.23</b>	17.91	5.19	<b>2.41</b>

Table 3.7: Rendered flag sequence performance vs the number of bases for *IROS* and SIFT-based tracking

Figs. 3.38, 3.39, 3.40 and 3.41 show the reprojection, reconstruction, rotation and translation error results of applying the tracking for 7, 15 and 30 bases.

Fig. 3.38 depicts the reprojection error for both tracking proposals. There is no difference among the different traces for SIFT-based tracking except for the peaks. The estimations using 7 bases are smoother. As the number of bases increments, the number of peaks is higher. Regarding the *IROS*-based tracking, the difference among traces is more evident due to the poorer tracking quality. In the middle the *loss* graph could be found (topic addressed in the following section). Similarly to the SIFT-based tracking,

the traces are very similar, although the one with 30 bases is, without doubt the worst one. When the tracking is stable there are little differences between 7 and 15 bases.

The differences are clearer analyzing the 3D error, in Fig. 3.39. The trend is, as the number of bases increases, the error also increases for both tracking systems. This trend is also followed in the rotation (Fig. 3.40) and translation (Fig. 3.41) errors.

Even the matching filters outliers, noise and considers visibility, it cannot filter all of them completely, and the model estimation algorithm tries to match this source of error with shape and pose changes. The more degrees of freedom added the worse, as the algorithm has more degrees of freedom to choose when modeling the error.

We can conclude that, with bad quality tracking the trend is reversed. Regarding a high quality matching, like SIFT, adding more degrees of freedom affects positively the 2D error and the translation errors, but the 3D estimation and rotation errors are degraded. The number of bases should be enough to proper model the deformation but not too low that the model is not complete. That is the reason because 15 bases were chosen as optimum in this thesis.

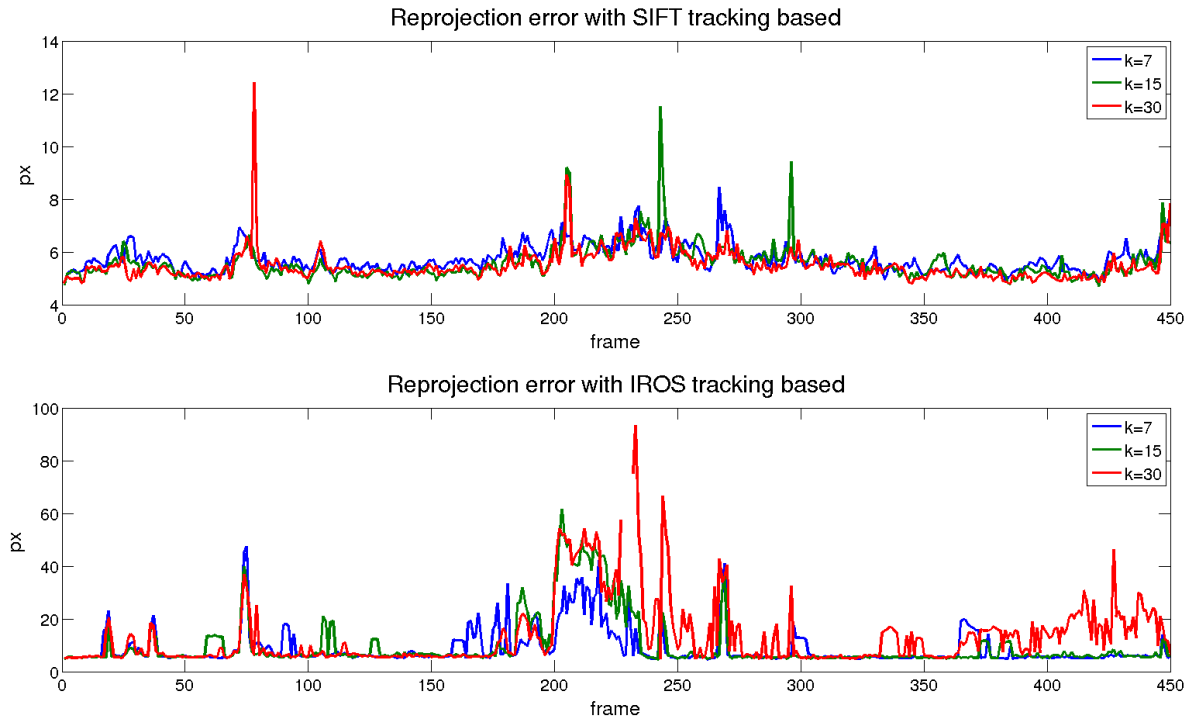


Figure 3.38: Flag sequence 2D reprojection error over time varying the number of bases for *IROS* and SIFT tracking.

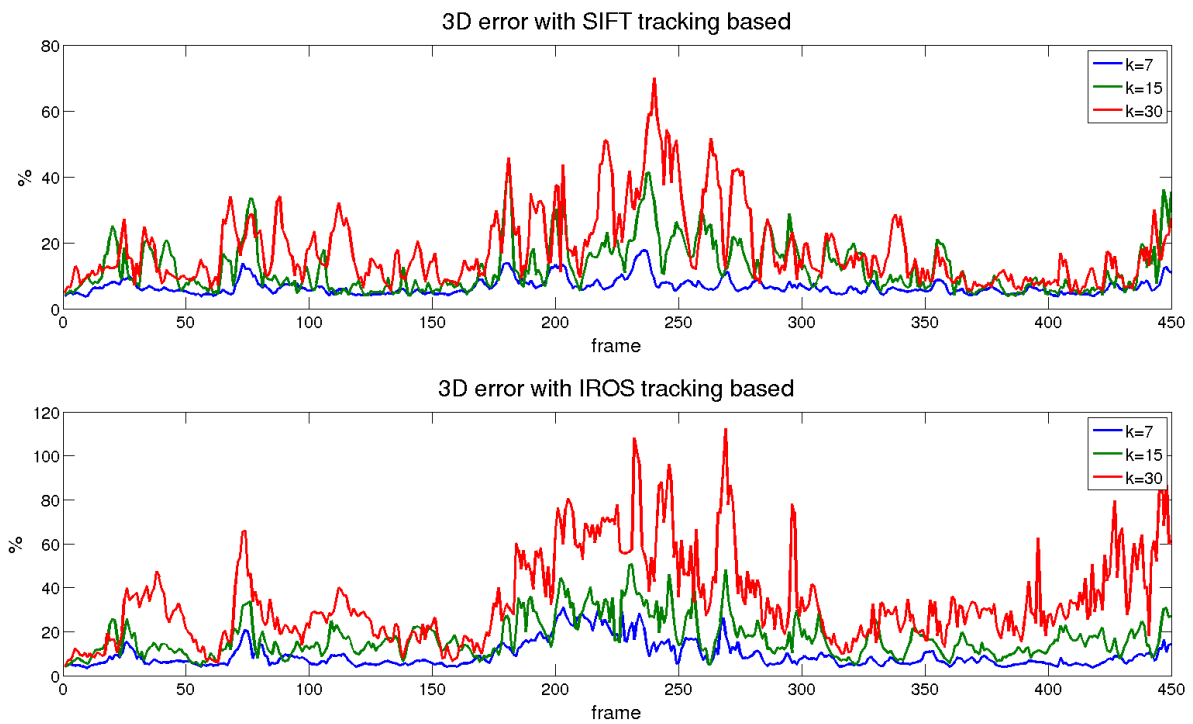


Figure 3.39: Rendered flag sequence 3D reconstruction error over time varying the number of bases for *IROS* and SIFT tracking.

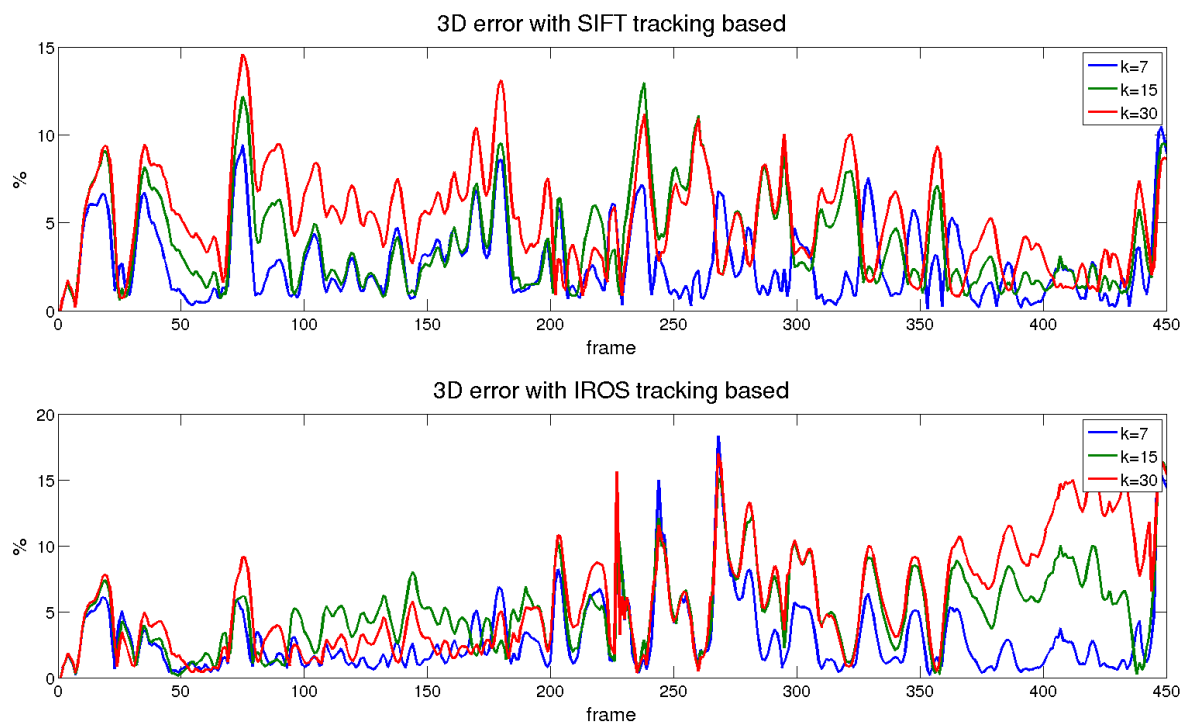


Figure 3.40: Rendered flag sequence rotation error over time varying the number of bases for *IROS* and SIFT tracking.

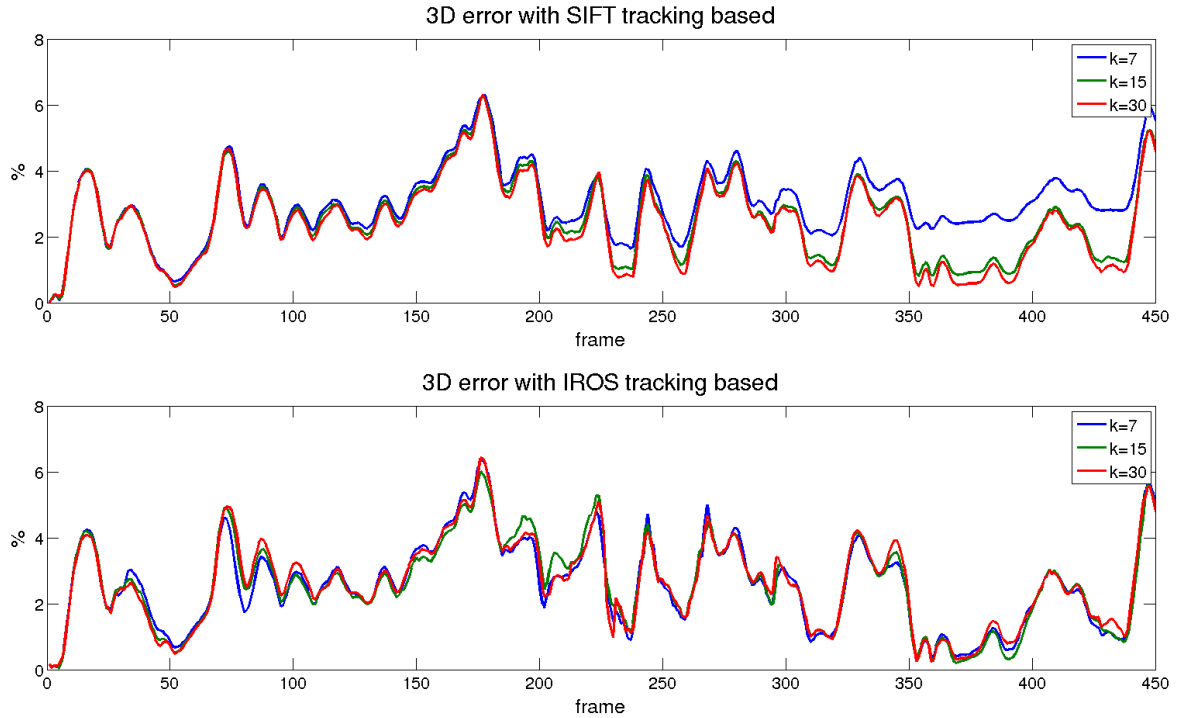


Figure 3.41: Rendered flag sequence translation error over time varying the number of bases for *IROS* and SIFT tracking.

### 3.2.2.3.4 Performance evaluation based on tracking loss recovery

Real tracking systems have to face losses in a natural way and after its detection, then a re-localization must be carried out. On a rigid system as PTAM, the only parameter to be varied is the pose, as the map is static. PTAM implements the active re-localization algorithm described in [Williams et al., 2007], which looks for 2D correspondences on the the image and actively tries to find a rotation with the currently feature points found on the image that best adapts the map points.

The active search algorithm is designed for rigid map scenarios, assuming the map static. In this section we test what happens if this algorithm is executed with a non-rigid sequence. The reconstructed shape before the tracking gets lost can be inexact, so it can be convenient to change not only the pose, but also the shape. Another more conservative approach can be preserve the non-rigid shape and run the standard active search algorithm for only the pose.

To check these approaches, the following setup is proposed:

1. The *IROS* tracking is tested as we present in [Bronte et al., 2014]. As mentioned before, this tracking procedure has the limitation of not correctly detecting points

further than 30 pixels. To fair comparison, the SIFT tracking searching area is limited to 30 pixels.

2. Regarding the global configuration, the tracking quality thresholds have been chosen to have enough number of points to start the active search algorithm and to have enough intervals of lost tracking for both approaches.
3. The tracking is considered lost if the sufficient amount of track with respect (15% in our case) the total are not detected in 3 consecutive frames. The transition between the states *lost* and *recovered* is immediate. Once the amount of visible tracks is above the threshold, the tracking is considered recovered.

In the top subfigure of Fig. 3.42 the tracking quality of the system is shown as a binary signal (tracking OK, tracking loss). The first row represents the SIFT tracking only varying the pose, the second varying pose and shape, the third the *IROS* varying pose and the forth varying pose and shape. The quality signal for the SIFT approaches (the first 2 rows) are identical, tracking loss are higher for *IROS* approach than for SIFT one.

In Fig. 3.42 the methods are compared for easier temporal comparison. We can conclude that both approaches are the same for SIFT (first row). Varying the shape and the pose has a positive effect on *IROS* tracking (second row). SIFT vs *IROS* comparison (third and fourth rows) shows that SIFT works better for both pose and pose+shape cases.

Focusing on the behavior over time and how it affects the errors, results are shown in Figs. 3.43 (reprojection), 3.44 (reconstruction), 3.45 (rotation) and 3.46 (translation) and tracking recovery has been added. The tracks are divided in section depending on the tracking quality (OK vs loss). For the tracking loss sections, pose recovery is drawn in red and pose and shape recovery in magenta. For the tracking ok sections it is necessary then to distinguish between the two recovery options because the tracking is different depending of the recovery applied in the previous tracking loss sections. If pose recovery has been applied the track is drawn in blue and if pose and shape was the option the track is drawn in green. The tracking with no recovery in the whole sequence is drawn in black.

Regarding the 2D reprojection error (Fig. 3.43), when the tracking is good there is not a high difference between the two proposals (blue and green) except for error peaks (in blue) that are produced because an inadequate shape from the previous loss sections is some trying to be rotated. For tracking loss section (red and magenta) errors are higher. Among the two, the best option is the pose and shape update. In the comparison SIFT vs *IROS* the former one gets less errors. SIFT gives good enough results without applying the recovery, similar to the ones obtained applying it with shape and pose. *IROS* approach needs any of the methods, preferably the pose and shape one, as, otherwise the

drift is accumulated.

Looking at the good tracking sections, for SIFT, there is no difference among the 3 proposals. With respect to the 3D error it can be seen that there is also not drift accumulated. Regarding the *IROS* approach, it can be seen that the reprojection error coming from pose (blue) is better than pose and shape (green). With respect to the 3D error, the pose and shape approach is much oscillating, but it gets faster lower values than the blue trace. Not applying recovery on *IROS* is crucial as the drift is accumulated in both 2D and 3D errors.

The last aspect of tracking recovery is the pose, composed of rotation (Fig. 3.45) and translation (Fig. 3.46). The recovery options taken in the tracking loss sections affects the evolution of these parameters. For rotation each option gives different evolution but the mean error gives more or less constant and in the same magnitude order than for no recovery option. The translation is less affected by the recovery option reaching a light improving for the only pose update regarding the no recovery option. The effect is higher in *IROS* than in SIFT.

We can conclude that, if the tracking algorithm is limited, such as in the *IROS* approach, it is recommended to change both the shape and the pose during the active search phase in a recovery process. If the tracking provides enough quality, as the presented in SIFT, the procedure can be disabled or, in the worst cases, the shape does not need to be changed, which results in an speed-up the computation time, as the shape does not need to be re-estimated.

As a final note we indicate that the algorithm has been applied to the Rendered flag sequence results by default so as to obtain a stable tracking with *IROS* to be able to compare with the rest of descriptors.

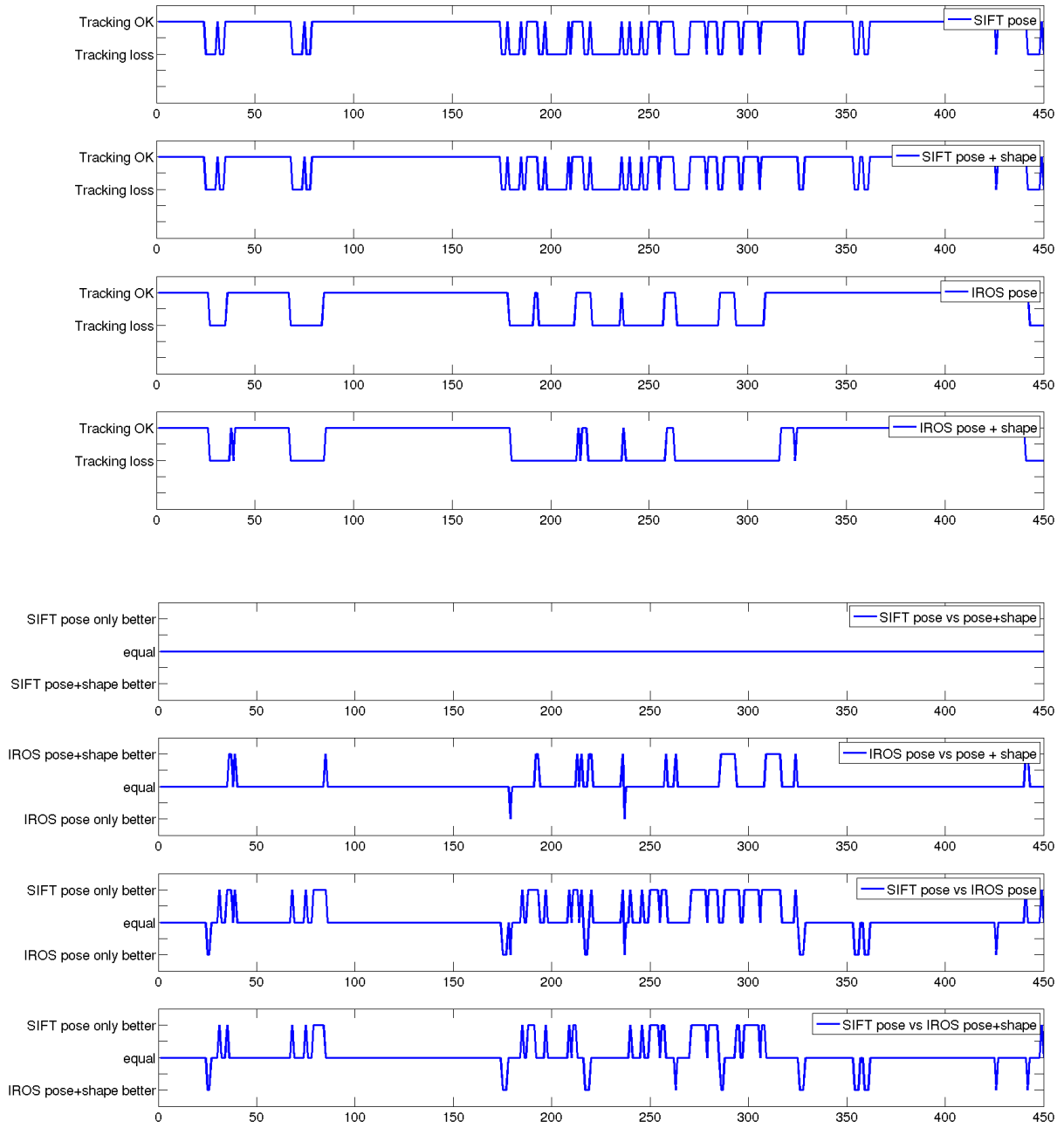


Figure 3.42: Rendered flag sequence analysis of temporal tracking quality for different descriptors. At the top, the absolute tracking quality values. At the bottom, the relative values, comparing a method against others

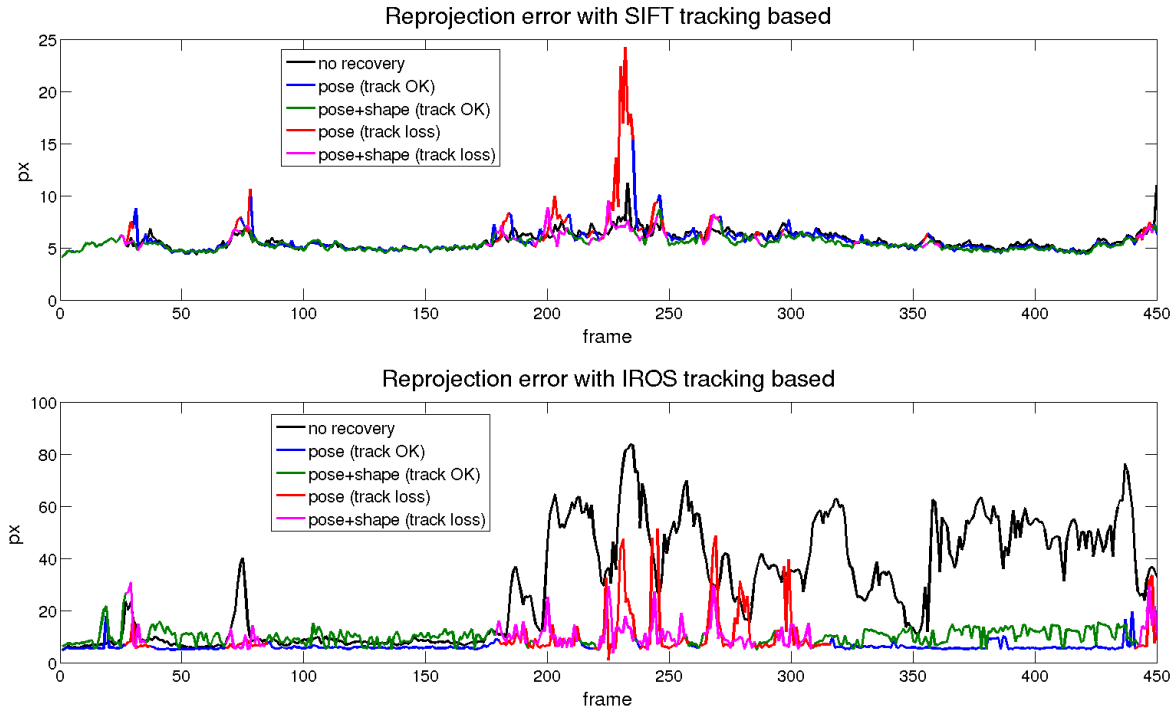


Figure 3.43: Rendered flag sequence temporal 2D error analysis for tracking loss. The first subfigure depicts the 2D reprojection error for the SIFT approach and the second one for the *IROS* approach. The losses are indicated with a different color: red (pose updated only) and magenta (pose and shape are updated).

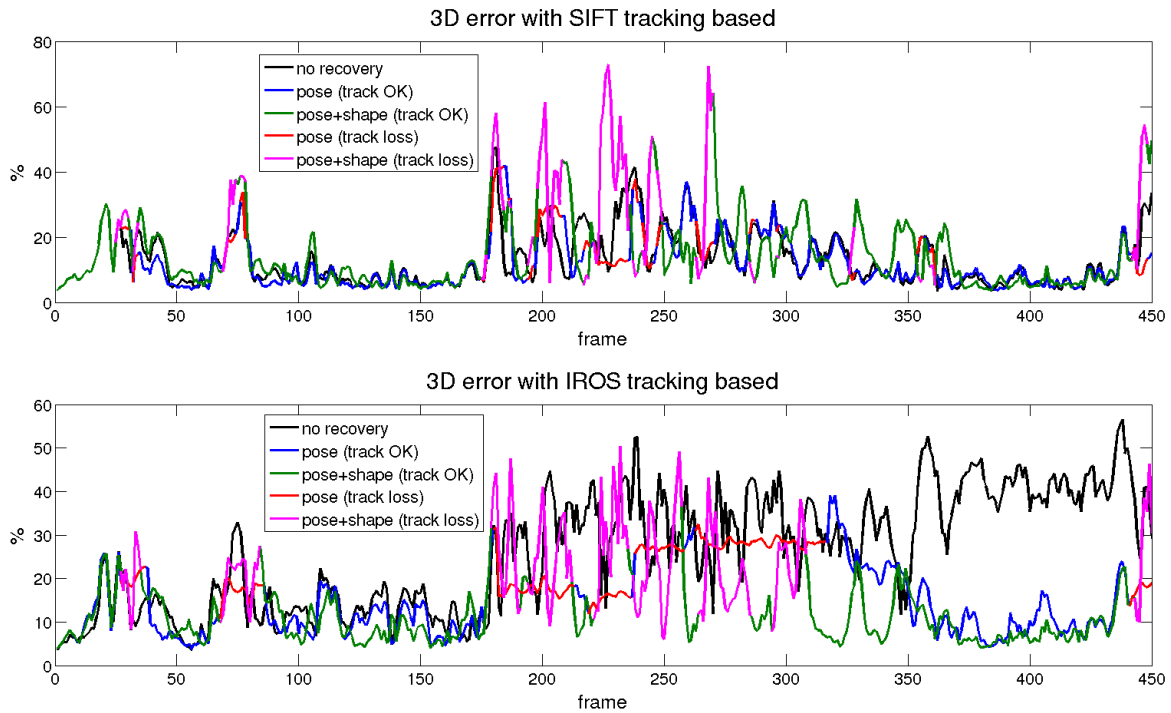


Figure 3.44: Rendered flag sequence temporal 3D reconstruction error analysis for tracking loss. The first subfigure depicts the 3D reconstruction error for the SIFT approach and the second one for the *IROS* approach. The losses are indicated with a different color: red (pose updated only) and magenta (pose and shape are updated).



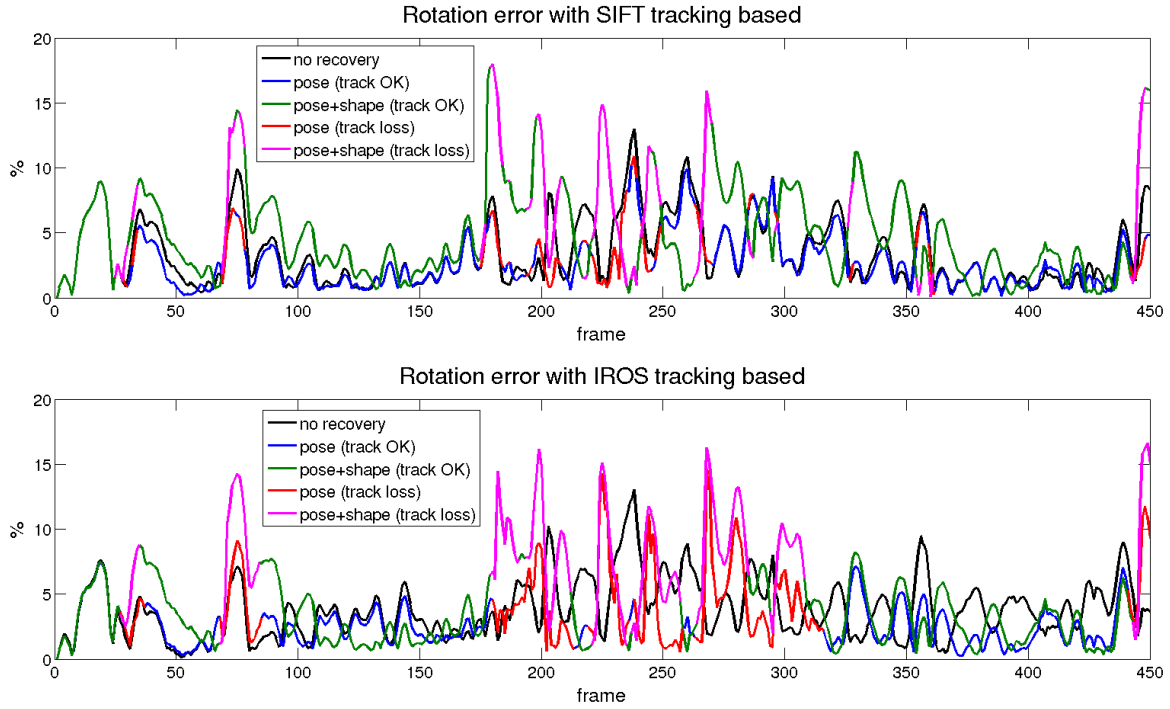


Figure 3.45: Rendered flag sequence rotation error analysis for tracking loss and different descriptors. The first subfigure depicts the rotation error for the SIFT approach and the second one for the *IROS* tracking based approach. The losses are indicated with a different color for each of the approaches compared, red (pose updated only) and magenta (pose and shape are updated).

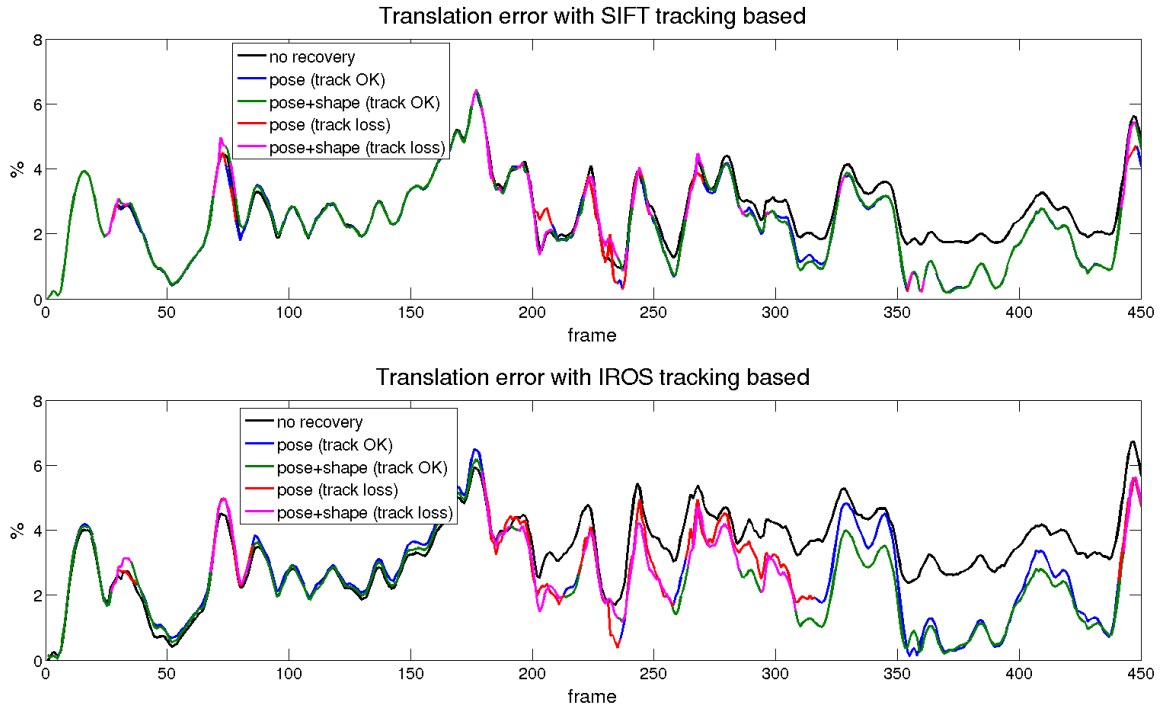


Figure 3.46: Rendered flag sequence translation error analysis for tracking loss and different descriptors. The first subfigure depicts the translation error for the SIFT approach and the second one for the *IROS* tracking based approach. The losses are indicated with a different color for each of the approaches compared, red (pose updated only) and magenta (pose and shape are updated).

### 3.3 Conclusions

This chapter has presented a real-time model-based deformable reconstruction algorithm. It holds a good trade-off between speed and quality. The presented approach uses as base PTAM, a well known SfM approach of the state-of-the-art for rigid objects able of processing its tracking and mapping threads in real time. We have proposed bypassing the mapping thread to insert a non-rigid model and adapting the tracking processing for non-rigid objects, maintaining the real-time constrain.

Our tracking proposal has been demonstrated to be comparable in accuracy to some of the most representative NRSfM current works when run on the same conditions, and it has successfully been scaled to deal with challenges as visibility, outliers and noise, which are not tackled by majority of the state-of-the-art references because they assume the tracking is solved.

Furthermore, to improve estimations over real images, the visual processing of the tracking thread was improved by adding feature descriptors. SIFT has demonstrated to be the best descriptor in terms of reprojection and 3D reconstruction error, although it is very slow to comply with real-time constraints. Therefore, AKAZE was chosen because it is the second best candidate in error performance and the first one in processing times.

Spatial and temporal smoothness is also tested. A combination of the two priors helps to reduce the flickering, which is noticeable on the 3D reconstruction error, and also stabilizes the estimated trajectory.

Our tracking algorithm provides a relocalization procedure in case the tracking gets lost, in the same way that PTAM does for rigid objects. From the authors knowledge, this is the first time that a recovery strategy is used for non-rigid tracking. It has been demonstrated that for tracking loss periods, changing the shape during the active search relocalization can be useful for a faster recovery, providing better performance.

## Chapter 4

# Sequential keyframe-Based Non-Rigid Mapping

As was mentioned in Chapter 2, the main drawbacks of existing statistics-based Non-Rigid Structure from Motion (NRSfM) method are their sensitivity to outliers and missing data and their high computational complexity.

This chapter proposes a sequential NRSfM solution that is able to cope with real data-association and outliers. In addition, it has computational complexity compatible with real-time without the need of a GPU. The proposed method is based on the Parallel Tracking And Mapping (PTAM) philosophy of dividing the problem into two parallel processes:

1. *Tracking thread:* given a shape model, this process computes the deformed pose of the object on each new frame of the sequence. It is based on the tracking method described in Chapter 3 and it's able of performing data association, detecting missing data and identifying outliers.
2. *Mapping thread:* takes a window of keyframes, selected from the Tracking thread, and computes a new shape basis. This process use a batch NRSfM solution for updating the shape-model from keyframes.

Dividing NRSfM into parallel tracking and mapping has several advantages:

- The tracking thread is fast, robust and keeps the reconstruction and camera pose up to date.
- Reprojection error in tracking is a cue to detect new keyframes. They represent novelty that should be included in the shape model.
- Outliers can be labeled as missing data on each keyframe. This is important as most of NRSfM methods are not resistant to outliers but some of them decently cope with missing data.

- The mapping thread is invoked only when not modelled deformations are detected. This keeps the number of keyframes small and under control. Model refinement is thus not growing exponentially with time. We use [Paladini et al., 2009] as the base of this method.

## 4.1 General architecture

The general architecture of state-of-the-art rigid Structure from Motion (SfM) systems, such as PTAM [Klein and Murray, 2007], Dense Tracking And Mapping (DTAM) [Newcombe et al., 2011b], Kinect-Fusion [Newcombe et al., 2011a], Dynamic Fusion [Newcombe et al., 2015], and other similar SfM approaches, decouples mapping from tracking.

In our proposal, the mapping thread is based on a batch NRSfM solution that takes as inputs a set of keyframes, selected from the tracking thread, and provides the new low-rank shape model, needed for reconstruction in the tracking thread.

The general architecture of the mapping thread is shown in Fig. 4.1.

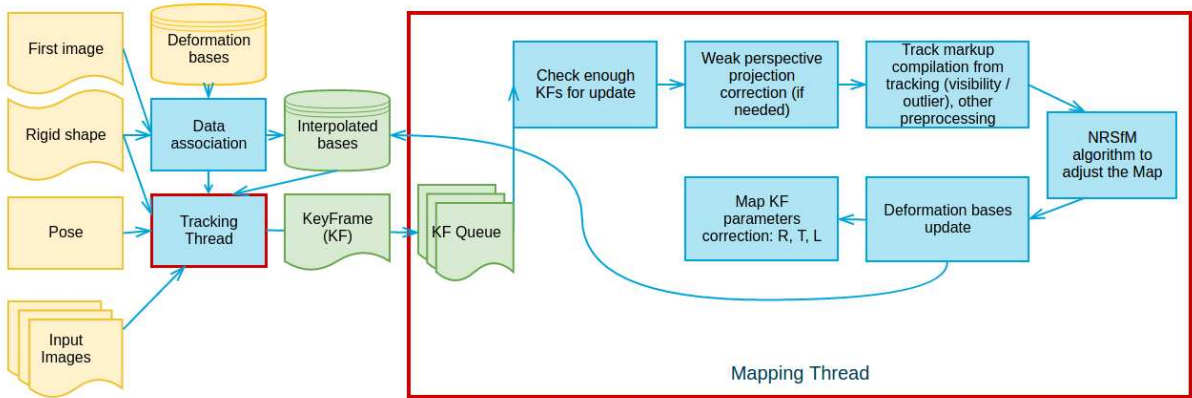


Figure 4.1: Architecture of the main mapping thread modules

The tracking thread block was detailed in chapter 3, in Fig. 3.1. The pipeline of the mapping thread consists of the following steps:

- The mapping thread has a queue of keyframes to be processed, which is updated from the tracking thread when the reprojection error is above a threshold. When the queue has enough keyframes NRSfM is invoked.
- As our NRSfM core works with the orthographic camera, image points in the keyframes are normalized with the average depth of the shape in the camera's reference, computed from the tracking solution (weak perspective). This correction is important as in the perspective camera the object has different scales for different

average depths whereas in the orthographic camera the average depth is not recoverable.

- The tracking thread detects outliers and handles missing data. This information is included in the queue of keyframes. In batch NRSfM methods, outliers are problematic. We mark them as missing data before invoking the batch NRSfM core.
- The NRSfM algorithm is run using the points included in the keyframes as inputs excluding outliers. It returns the reconstructed shape of the object on each keyframe.
- The new shape basis is constructed by using Principal Component Analysis (PCA). Shape coefficients and camera pose are updated in the keyframes and the new basis are sent to the tracking thread.

This pipeline strategy does not imply any blocking of the tracking thread that keeps computing the reconstruction while the result of the mapping is completed.

This approach is inspired in the sequential part of [Paladini et al., 2010], but the main difference is that not all the frames are exhaustively evaluated, but only the acquired keyframes from the tracking. Our approach also allows not to be stuck to a fixed / incorrect / degenerated deformation model, very common in sequential methods such as [Paladini et al., 2010].

The changes due to a single keyframe addition are low although among several keyframes they are enough to provide a noticeable correction on the model for unseen deformations. In addition, computing model updates in sets of keyframes instead of doing it each time a keyframe is added, reduces CPU load of the mapping thread.

To sum it up, a pseudo algorithm of the mapping thread is depicted in Algorithm 4.1.

## 4.2 Sequential modeling validation proposal

In our proposal, the selected NRSfM algorithm to compute the new bases from the incoming tracks is Metric Projection (MP) [Paladini et al., 2009]. Despite not being the best state-of-the-art algorithm nowadays, it is fast and copes with missing data. Processing time can be improved as the method can be easily parallelized.

This NRSfM algorithm works with the orthographic camera while our system and the tracking thread assumes the perspective camera. As mentioned before, a solution to alleviate this problem is to normalize the image coordinates using the scene’s average depth, recovering approximated image coordinates of the orthographic camera.

This is equivalent of using the weak-perspective camera model, where the points are projected with the orthographic camera at the plane  $z = z_{mean}$ , where  $z_{mean}$  is the scene’s average depth in the camera reference. This plane is then projected with the perspective

**Input:** Pose, rigid shape, initial set of *deformation bases*  
**Output:** Improved *deformation bases*, 3D reconstruction and pose for each frame  
Initialize the algorithm giving the initial set of bases, a rigid shape, and the initial pose pose;  
start the tracking process;  
start the mapping thread;  
tracking thread();  
**while** *not at the end of the sequence* **do**  
    Map Tracking();  
    Measure reprojection error;  
    **if** *repr err > threshold* **then**  
        send keyframe to Mapping thread;  
mapping thread();  
**while** *not Stop* **do**  
    check keyframe Queue;  
    **if** *There are enough keyframes on the Queue to update the whole map* **then**  
        correct the tracks with weak perspective projection approximation;  
        compile visibility and outlier information from each point;  
        call NRSfM core algorithm;  
        compute normalized bases from reconstructed shapes;  
        update shared memory with the tracking data;  
        correct keyframes poses and coefficients of the whole map;

**Algorithm 4.1:** Pseudo algorithm for the proposed NRSfM update proposal

camera. This is an affine projection model that accounts for the scale change that is visible when the objects are placed at different depths from the camera.

The perspective camera projects a point with coordinates  $(x, y, z)$  in the camera reference as follows:

$$U = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \alpha \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \begin{pmatrix} \frac{x}{z} \\ \frac{y}{z} \end{pmatrix} \quad (4.1)$$

where  $\alpha, f_u, f_v, u_0$  and  $v_0$  are the intrinsic parameters. In the weak-perspective camera the same point is obtained as follows:

$$U = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} + \alpha \begin{pmatrix} f_u & 0 \\ 0 & f_v \end{pmatrix} \begin{pmatrix} \frac{x}{z_{mean}} \\ \frac{y}{z_{mean}} \end{pmatrix} \quad (4.2)$$

where  $z_{mean}$  depends on the scene. Given image points whose projection can be approximated with the weak-perspective camera, the following transformation gives a new set of image points compatible with the orthographic camera:

$$U' = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{(u-u_0)*z_{mean}}{\alpha*f_u} \\ \frac{(v-v_0)*z_{mean}}{\alpha*f_v} \end{pmatrix} \quad (4.3)$$

When the shapes are computed using orthographic based algorithms the size of the retrieved shapes are given up to scale. Therefore, the 3D bases are normalized so each of the shapes has a norm equal to 1 when they are loaded or modified by the re-computation, so to preserve the sizes of the estimations on the tracking.

An aspect that must be taken into account for real operation is that, when the bases are re-estimated, the set of keyframes stored in the map must be re-computed, that is the set of coefficients must be recomputed and also the set of poses (the NRSfM algorithms also provide this information even working with orthographic projection).

The orthographic projection matrix lies on the Stiefel manifold<sup>1</sup>. The full 3D rotation matrix is reconstructed by the cross product of the two rows of the Stiefel matrix (if the  $K^{-1}$  intrinsics matrix has not been previously taken out). An approximation of the translation vector is retrieved, although the depth must be corrected with the weak perspective projection with Weighted Least Squares (WLS).

## 4.3 Results

### 4.3.1 Experiment setup

In order to validate our proposal, we test the update of the shape-model to unseen deformations. Further implementation is needed to combine the mapping strategy with the tracking thread in a real application. The following procedure is performed.

The Flag sequence is divided into 8 equal slots. Our algorithm starts from a known but reduced model, obtained from a small number of frames, which is not likely to be valid for all the sequence, as it does not contain many of the deformations that will appear in future frames.

The initial pose is the ground truth. The rigid shape is the usually given one from the point-wise Flag sequence and the base shape are composed by the first slot of the sequence reduced by a subsampling of 1/5, giving the shapes of the frames 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 and 55. With this, a simulation of the minimum keyframe insertion interval on the mapping thread is set with a minimum interval of 5 keyframes (150 ms sampling time in a 30 ms real time). This also assumes that the tracking thread detected a reprojection error above the minimum defined. With this setup the keyframe queue size is set to 12.

The sparse ground truth shapes of the gridded 540 points are projected with the ground truth object pose with perspective projection, simulating the tracks obtained for those frames. Then, the weak perspective tracks are computed using the average depth of the ground truth shapes for those shapes. After that, and assuming all the tracks are visible for all the points, MP is run with the weak perspective tracks, obtaining the shapes for all the depicted frames. Finally, the bases are computed with the maximum rank indicated by the MP algorithm. If the algorithm does not converge due to singular matrices found

<sup>1</sup>From [Paladini et al., 2012]: The Stiefel manifold  $V_{k,m}$  may be viewed as the collection of all  $m \times k$  matrices whose columns form an orthogonal set. More precisely, the (real) Stiefel manifold  $V_{k,m}$ , is the collection of all ordered sets of  $k$  orthonormal vectors in Euclidean space  $\mathbb{R}^m$

during the factorization of the algorithm, the rank is lowered. For this case, the rank had to be lowered to 6 bases.

Using this setup, without any optimization, using 450 points, 12 frames and no missing data, the algorithm takes seconds to compute the whole set of bases. It is important to remark that no extra iterations are needed to improve the result as we have not missing data. The initialization step, the need of more iterations and more frames is an important factor to take into account.

With this setup, the tracking thread starts and the data association is performed on the provided input data, to transform the input gridded data to data directly related to the detected ones by means of the estimation of the barycentric coordinates, as explained in section 3.1.1. Once the association is done, it will be preserved and further used for the rest of the iterations (on the real operation, this association does not change and the points remain the same for all the sequence).

After the tracking algorithm yields its results, they are gathered and compared with the ground truth. The tracking data is retrieved and part of the process on the first step is followed, although the data source changes.

As the tracks are directly retrieved and aligned from the tracking, they do not need to be projected, but they are directly related to the 3D points. The weak perspective correction is again performed, but using the estimated 3D data from the tracking, just for the evaluated keyframes.

The visibility mask is filled up from the visibility flags and the outlier information of the tracking. To do the merging of the visibility information ( $\text{visible}=1$ ) with the outliers ( $\text{outl} = 1$ ), the outlier logic must be inverted,  $\text{MD} = \text{vis} \ \& \ \sim\text{outl}$ .

The MP algorithm is run again with this information and the bases are computed as in the previous step. Note that this time the shape is recovered from the bases that have already been interpolated, so the aspect is not the grid-like as the point wise Flag sequence. A sparse point cloud set of points, based on the detected ones on the first frame, is created as the data association was already performed on the tracking thread.

In order not to alter the normal initialization and data association of the tracking, and most important, to make both initialization ways compatible, as the data structure changes (from a gridded-like to a point to point one), an index guided setup is performed between the interpolated data and the gridded one. A look up table containing the indices of the interpolated points is stored and loaded when this direct assignment is done to the bases.

To preserve the data association in all the trials, the algorithm is run from the beginning all the time, so as to obtain exactly the same points on the first reference frame. With this strategy, the same 3D reference points can be obtained for all the trials and the test conditions can start in the same way, doing that the comparison can be performed also in the same way.



Even if it could seem the same approach, regarding the part that is running in the tracking thread, with respect to the initialization step, it is decided to move a step forward to be closer to any standard NRSfM approaches. The detected points by the tracking are the ones used to generate the bases, instead of the previously seen in chapter 3 where the grid-like shapes come from the ground truth.

It must to be pointed out that this interpolation directly depends on the type of feature used on the experiment, i.e. SIFT, SURF, KAZE, etc, as the feature points detected by each algorithm are different and have different properties. For the current setup we have chosen KAZE for its repetitiveness and stability, although any of the well known feature point descriptors could have been used for this experiment. AKAZE was first tested, but there were certain areas that were not fully covered by the features, so KAZE was finally chosen instead.

The minimum keyframe interval was 5 frames and the error threshold was low enough to accept all the frames into the mapping thread. This first approach was carried out because the tracking thread is implemented in c++, but MP code is available in Matlab. A more realistic implementation will require to improve these figures. For live tests the whole code should be compiled in c++ and integrated on the mapping thread.

When the missing data is included in the estimation with MP, the bases computation takes about two minutes for the longest trial (90 frames, 350 points). It is 4 times longer compared to the the results obtained when the model was obtained using ideal conditions. This difference is due to the needed iterations on the optimization process, not in the amount of data to process (number of points and frames).

#### 4.3.2 Test Sequence: Rendered flag sequence

The baseline to compare the results is generated with a model using MP algorithm from a set of synthetically projected set of all frames with the ground truth tracks of all the frames with the weak perspective corrected tracks, and generating the bases by using PCA. The rigid shape used to perform the initial data association is the usual gridded one. The proposal includes visibility and the outlier information merged to the missing data matrix of the MP algorithm. The tracks are the extracted ones directly compiled from the tracking thread.

The initial phase is carried out when the model used from the ground truth (gridded model) is used from frames 1 to 56 (the first slot of the sequence). The results from the baseline vs from the first created model with the first interval are depicted in Fig. 4.2, (the first two rows) for frames 1, 25 and 50.

The 3D reconstructions are depicted for the baseline and the reduced set of the bases on the first two rows of Fig. 4.6, for the same frames.

The rest of the sequence is analyzed, for the frames 60 and so on, for the baseline and

the generated bases, the screenshots are shown in Figs. 4.2, 4.3, 4.4 and 4.5. The 3D reconstructions for the rest of the sequence are analyzed for each of the intervals on each of the rows on Figs. 4.6, 4.7, 4.8 and 4.9

On Figs. 4.10, 4.11 4.12 and 4.13, the results over time are presented. The first blue trace represents the first generated model performance against the baseline. The results of the dynamically generated models in each time slot are represented in a different color. The discontinuities are the consequence of the model substitution.

Looking at the evolution of the 2D reprojection error (Fig. 4.10), the performance is similar, or sometimes better, compared to the baseline model. Regarding the 3D reconstruction error, the general trend is similar, although it depends on the time slot evaluated the error is greater or lower. During the 5th interval, the 3D reprojection error rises for both baseline and our approach, but the presented approach is more sensitive to this circumstance. Regarding the rest of the errors, a similar trend can be observed, being improved all the intervals the rotation error and most of them on the translation errors (except 7th and 8th).

The transition time, time where the new bases are being computed on the mapping thread, is not considered in the NRSfM process because it depends on the implementation and the selected algorithm. As the sampling time was also emulated, the substitution time was considered 0. As soon as the interval finishes, the new bases are available and the computation are performed with the new ones.

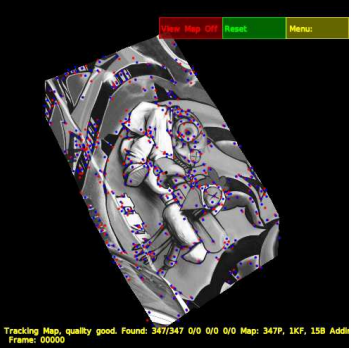
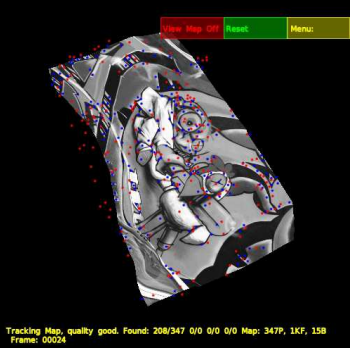
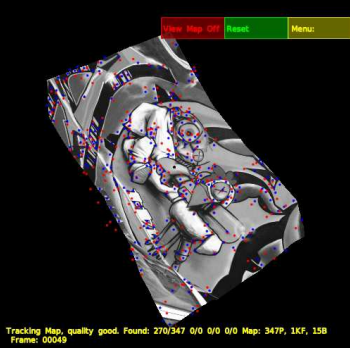
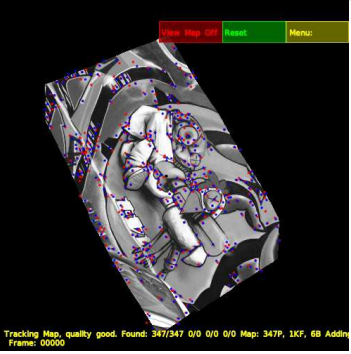
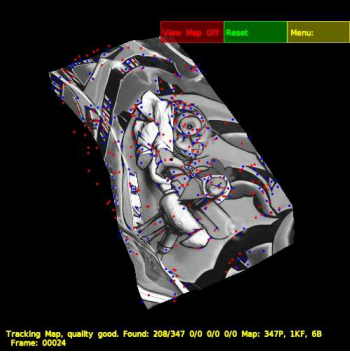
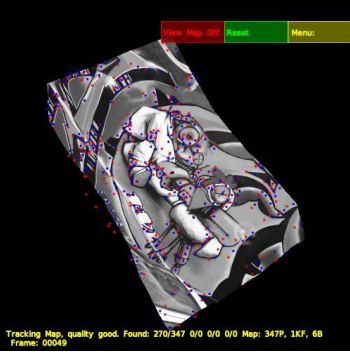
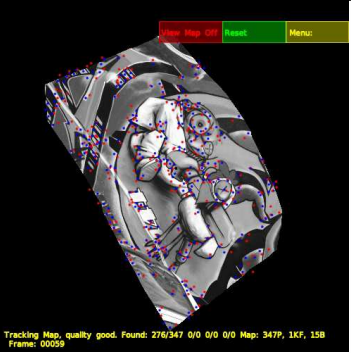
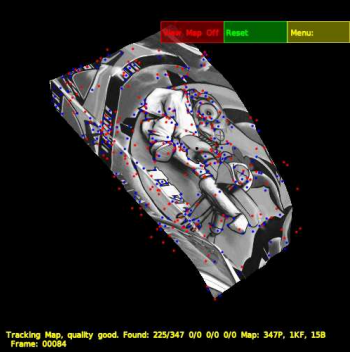
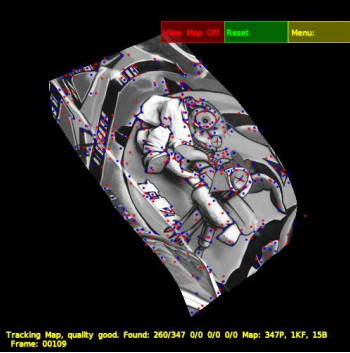
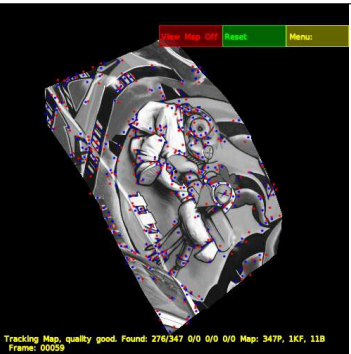
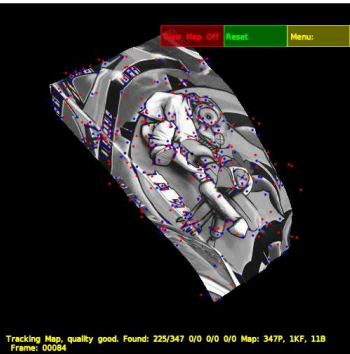
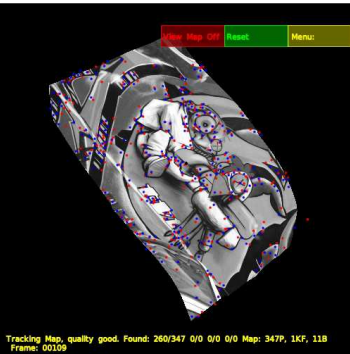
step	trial	f=1	f=25	f=50
1	base			
1	vis+out			
		f=60	f=85	f=110
2	base			
2	vis+out			

Figure 4.2: Screenshots for flag sequence processing (first fourth)

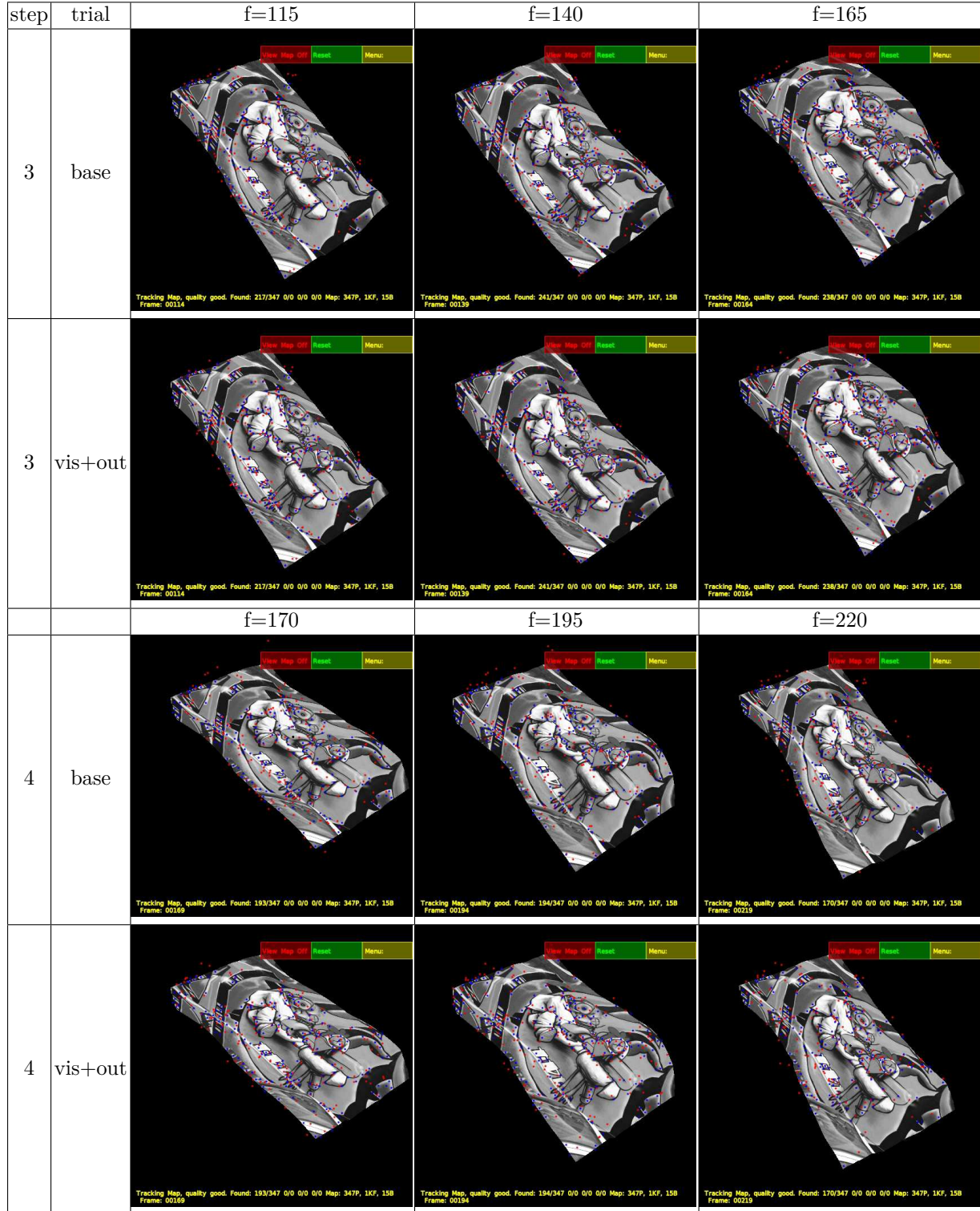


Figure 4.3: Screenshots for flag sequence processing (second fourth)



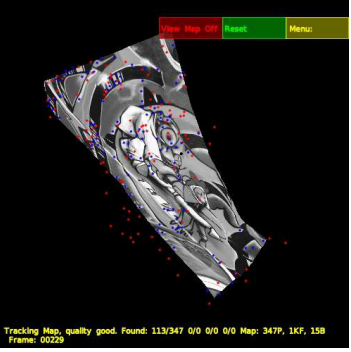
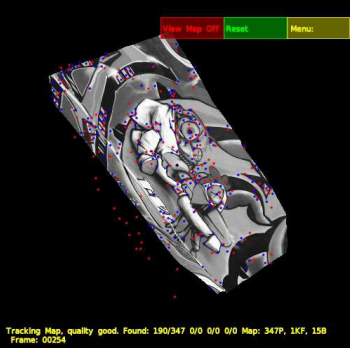
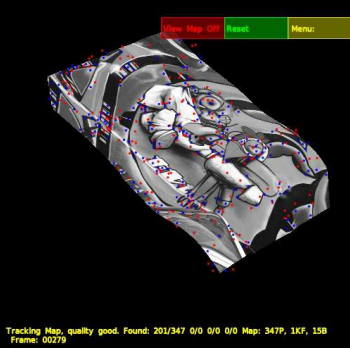
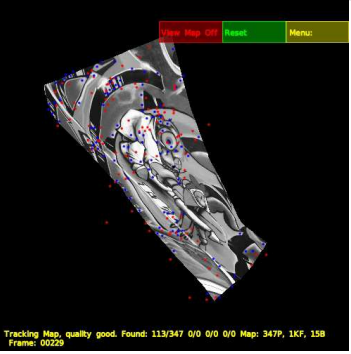
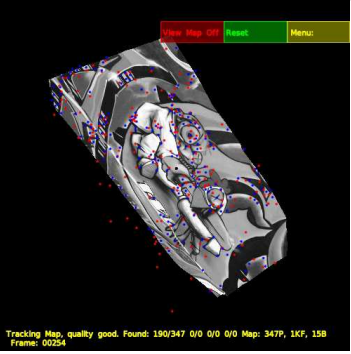
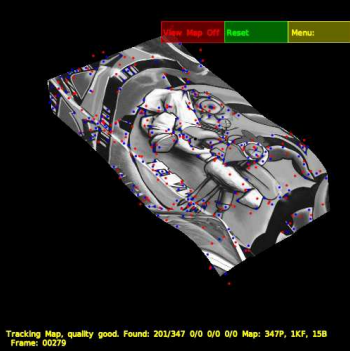
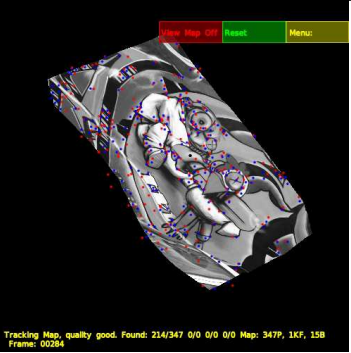
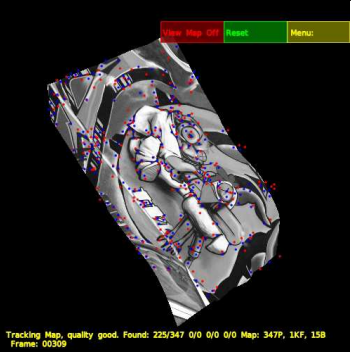
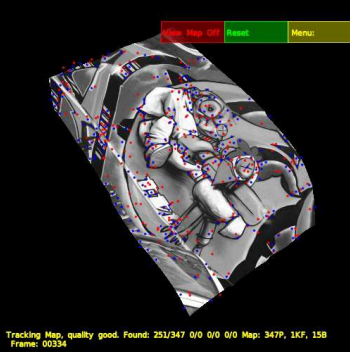
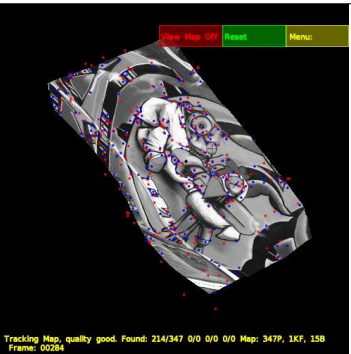
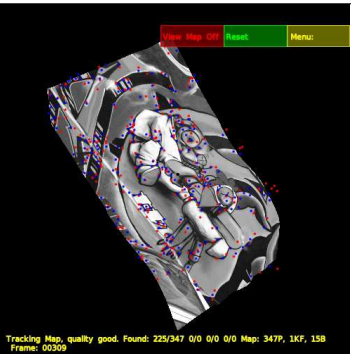
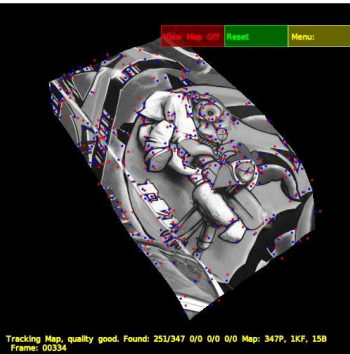
step	trial	f=230	f=255	f=280
5	base			
5	vis+outl			
		f=285	f=310	f=335
6	base			
6	vis+outl			

Figure 4.4: Screenshots for flag sequence processing (third fourth)

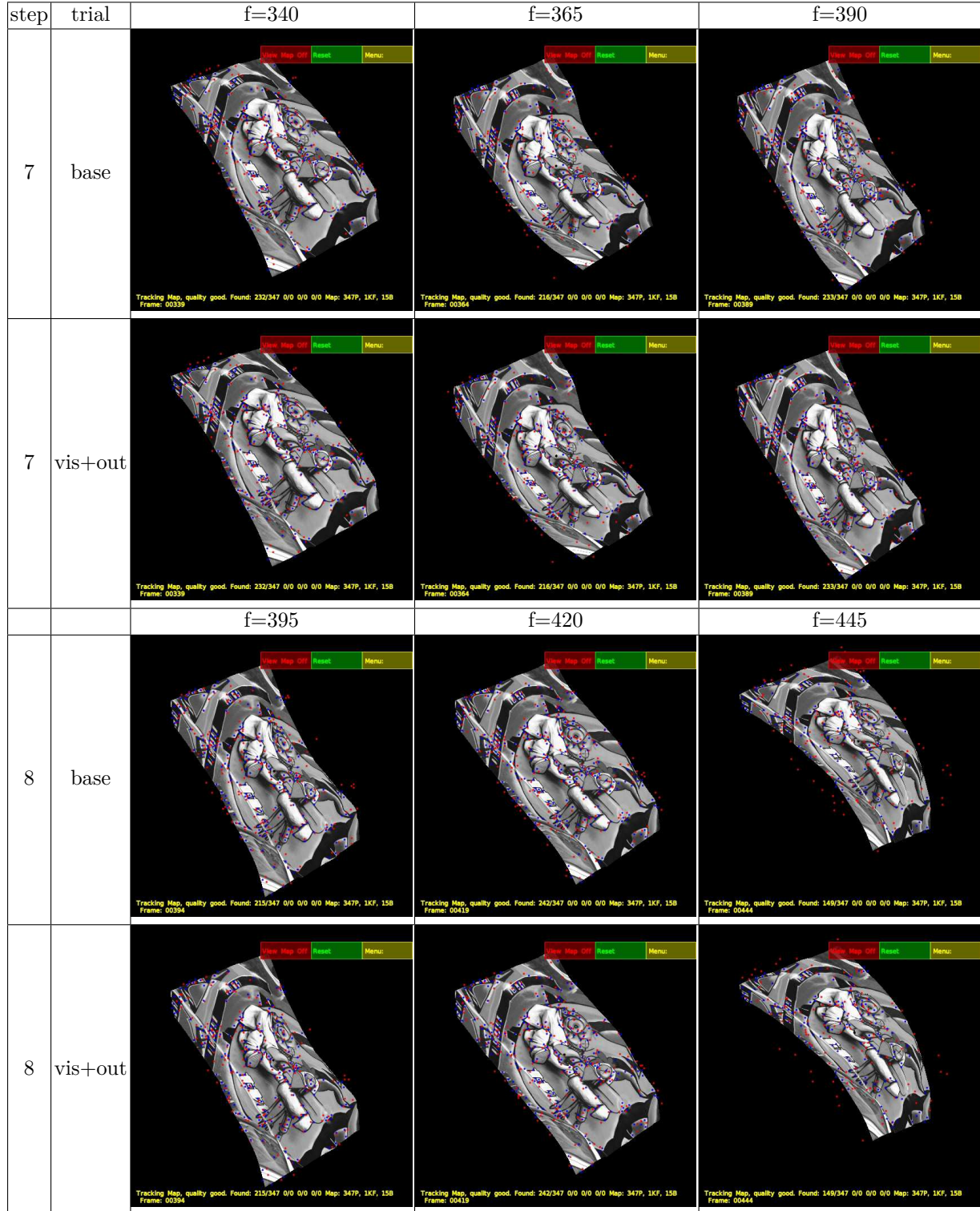


Figure 4.5: Screenshots for flag sequence processing (four fourth)

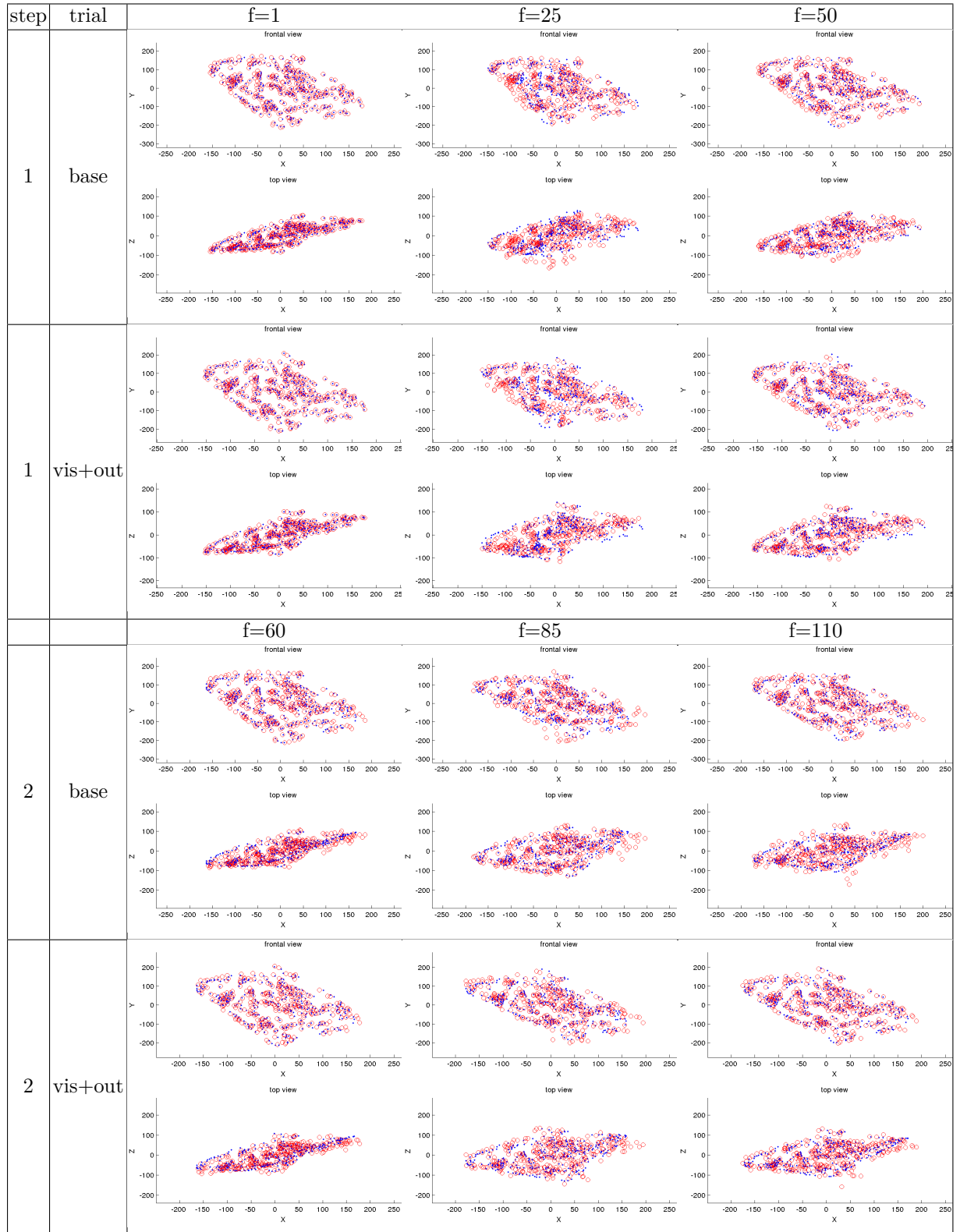


Figure 4.6: 3D reconstruction representations for flag sequence (first fourth)

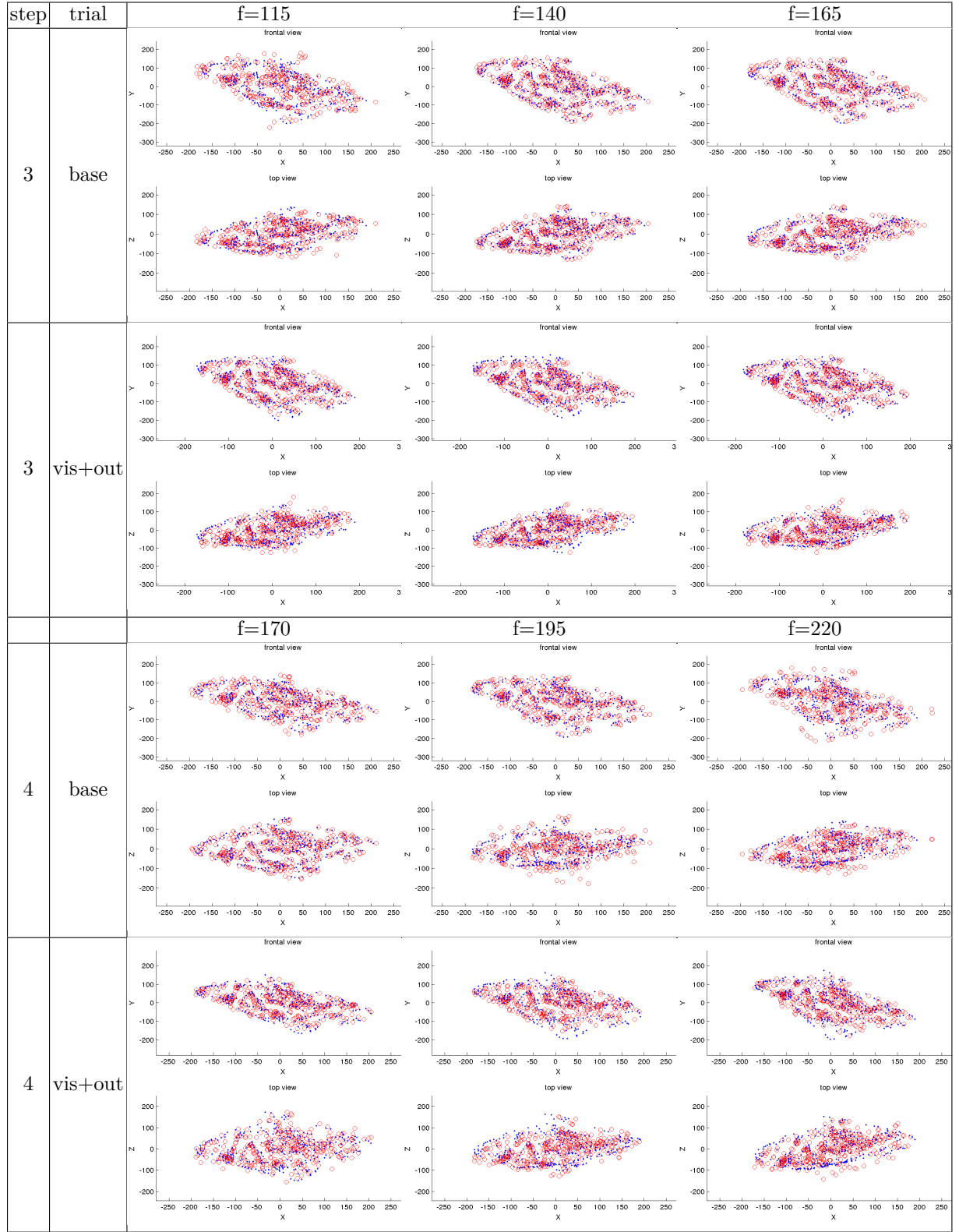


Figure 4.7: 3D reconstruction representations for flag sequence (second fourth)



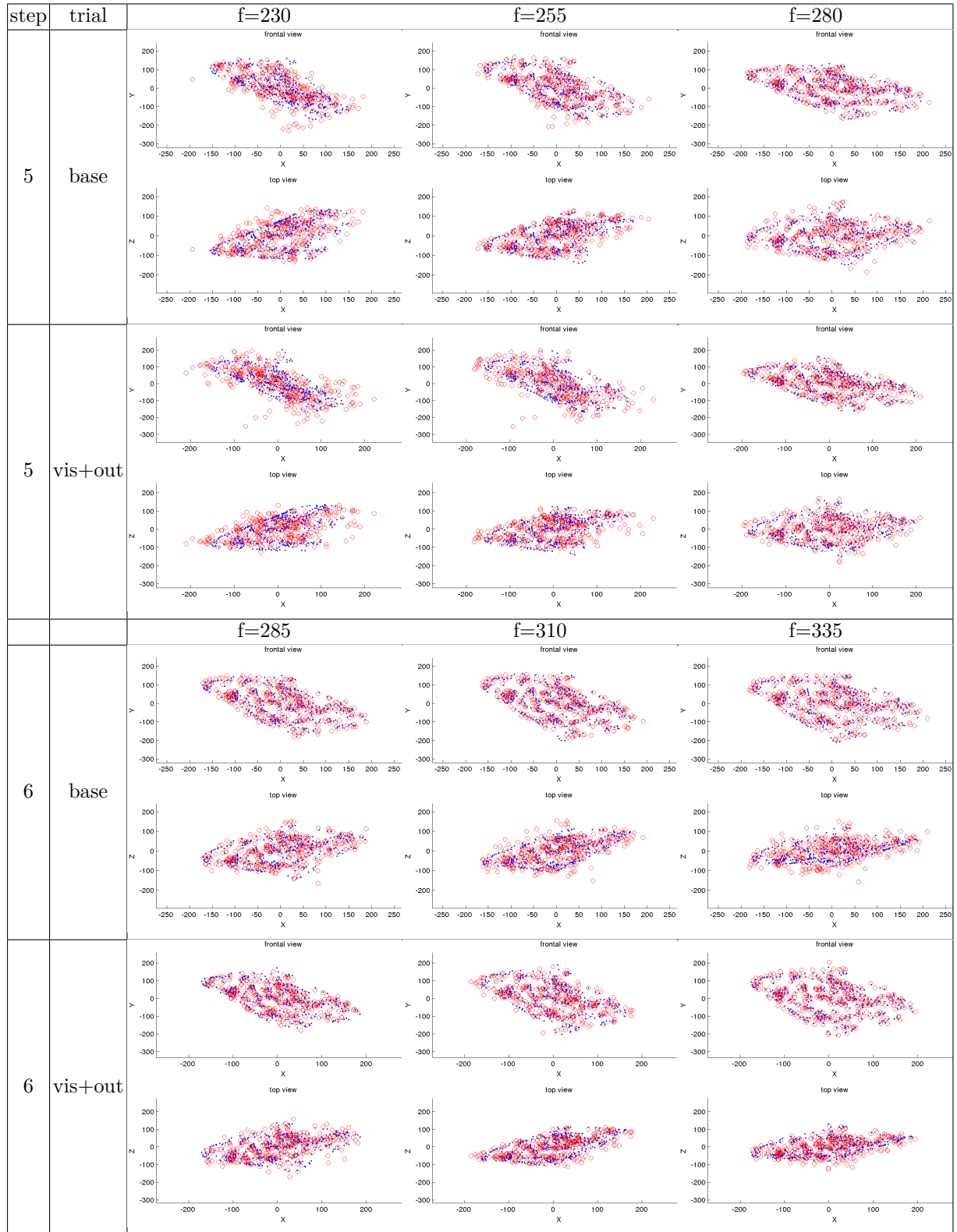


Figure 4.8: 3D reconstruction representations for flag sequence (third fourth)

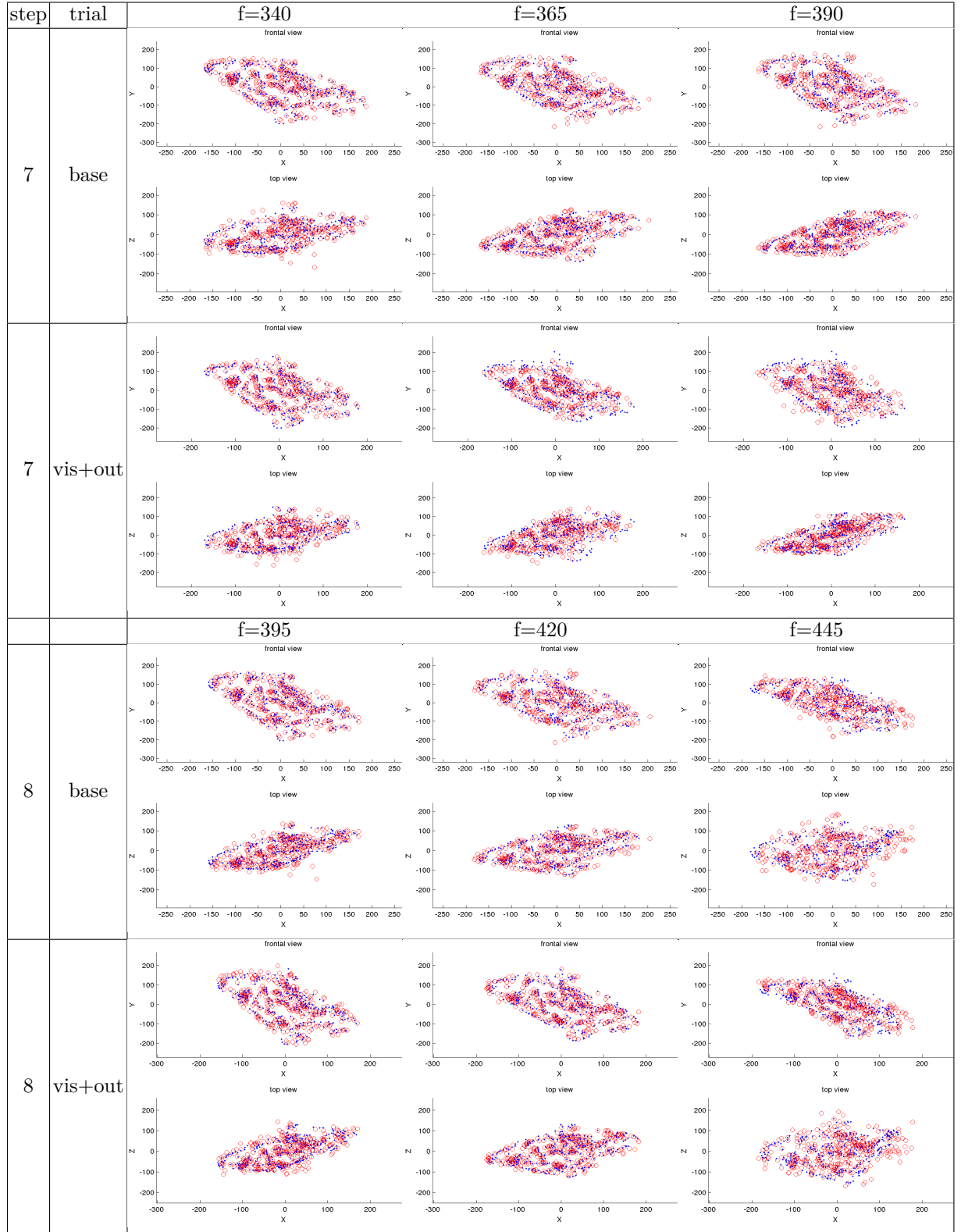


Figure 4.9: 3D reconstruction representations for flag sequence (four fourth)

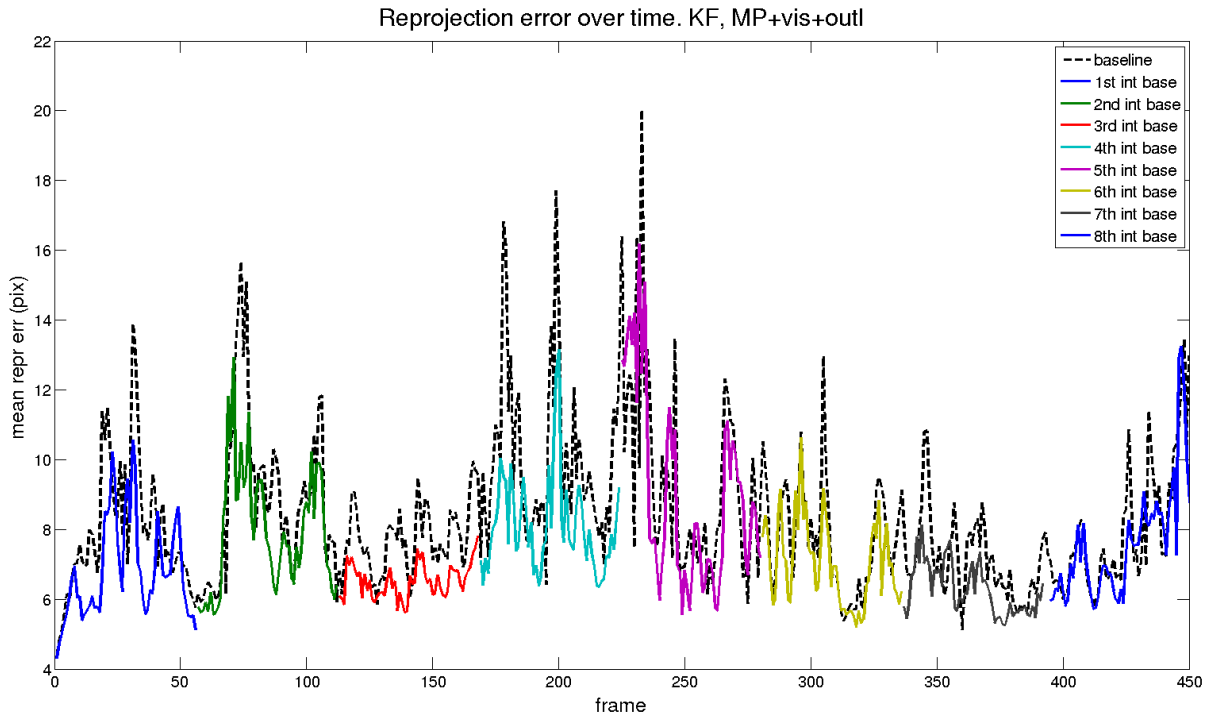


Figure 4.10: 2D reprojection error over time for visibility and outlier rejection when regenerating the bases. The approach taking into account visibility and outliers is shown compared to the baseline

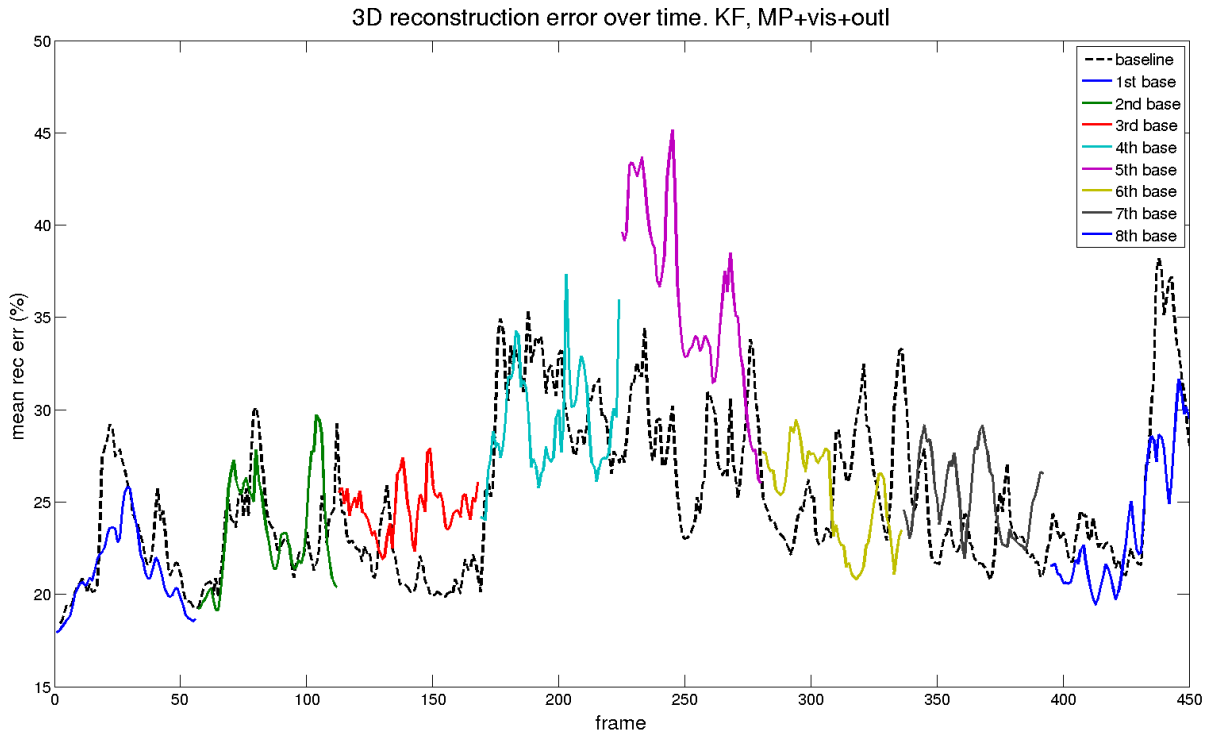


Figure 4.11: 3D reconstruction error over time for visibility and outlier rejection when regenerating the bases. Top subfigure, the approach taking into account visibility and outliers with the normalized 3D error. Bottom subfigure, the approach taking into account visibility and outliers without normalizing the 3D error

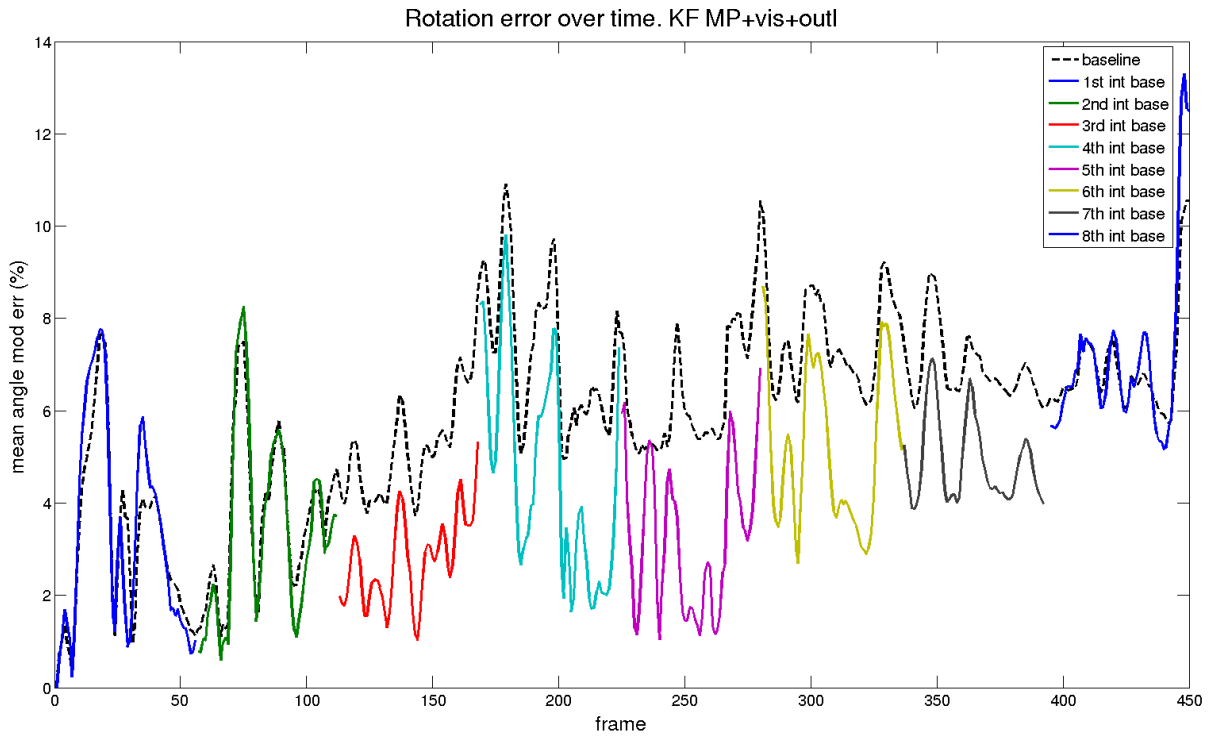


Figure 4.12: rotation error over time for visibility and outlier rejection when regenerating the bases. The approach taking into account visibility and outliers is shown

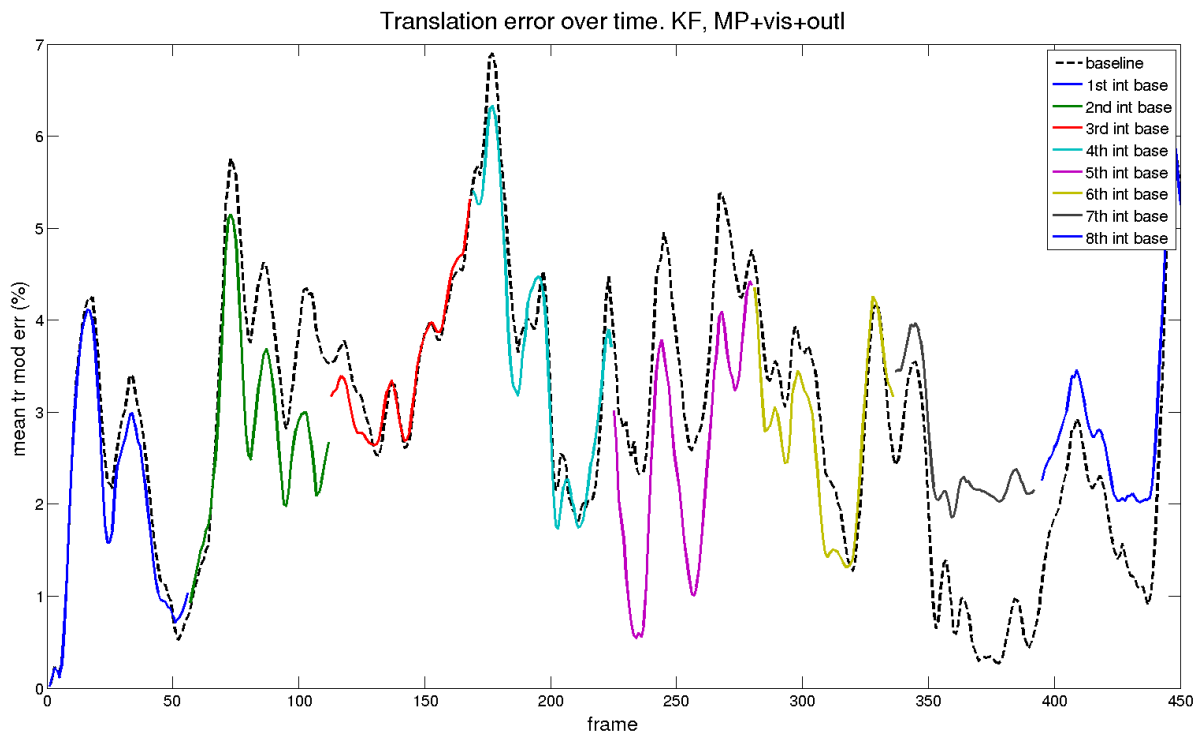


Figure 4.13: Translation error over time for visibility and outlier rejection when regenerating the bases. The approach taking into account visibility and outliers is shown

## 4.4 Conclusions

Regarding the sequential approach based on keyframes instead of doing a per frame reconstruction, it has been shown that similar results are obtained with the chosen baseline algorithm in the experimental results MP and the keyframe based one even the amount of information is substantially reduced because only some frames are taken for incrementing the base.

It has also been demonstrated that the efficient filtering of missing data using the information provided from the tracking thread is useful to obtain similar results as the getting with the ideal model.

Our proposal has been tested and checked that the coefficient upgrade after deformation base update is correctly performed for all the map keyframes.

Separating the tracking and the mapping processes in two threads makes the mapping so flexible that the NRSfM algorithm could be changed without a great impact in the rest of the system architecture except for the introduced processing delays inherent to the algorithm.



## Chapter 5

# Conclusions and Future Works

This thesis has contributed Non-Rigid Structure from Motion (NRSfM) research line accomplishing the task of reaching an implementation ready for real-time processing at the same time that obtains a good trade-off between processing time and reconstruction quality.

The proposal relies on a well known state-of-the-art Structure from Motion (SfM) algorithm, as is Parallel Tracking And Mapping (PTAM), which presents a parallel architecture to get a map and a camera pose tracking efficiently for rigid objects. We have modified the tracking thread of PTAM to do it compatible with non-rigid objects, holding its real-time constraints, over real images. The mapping thread has been substituted for a NRSfM method. The state-of-the-art methods focused on NRSfM opt for batch or sequential approaches but based on mono-threaded implementations, which are not scalable and very sensitive to outliers and missing data. We propose a sequential keyframe-based NRSfM solution able to cope with real data-association and outliers and with a computational complexity compatible with real-time.

### 5.1 Conclusions and contributions

In the next paragraphs we sum it up the main conclusions and contributions extracted from the two threads of our NRSfM system: tracking and mapping. First, we focus on the conclusions and contributions regarding the tracking of the non-rigid objects, assuming the model is known and fixed.

- The presented algorithm is able of performing a real-time tracking for non-rigid objects and supposes a good trade-off between speed and reconstruction quality regarding other approaches of the state-of-the-art.
- The full state estimation can be performed in an alternating scheme using an Expectation-Maximization (EM) algorithm, first computing the shape deformation

coefficients in closed form fixing the pose, and then computing the pose assuming the shape as rigid. This strategy decouples the estimation in two parts, easing implementation and improving convergence time, which contributes to the real-time constraint.

- Our proposal has successfully been scaled to deal with real visual processing challenges as visibility, outliers and noise handling, which only a few state-of-the-art works face, as most of them assume an ideal tracking of features.
- Unlike [Agudo et al., 2016a], in which only tens of points are handled due to real-time restrictions, this proposal scales to hundreds of points.
- We have faced detection strategies on real images. A comparison among different feature detectors of the state-of-the-art has been carried out. SIFT has demonstrated to be the best descriptor in terms of reprojection and 3D reconstruction error, although it is very slow to comply with real-time constraints, so the detector chosen for our proposal has been AKAZE because it is much faster and its errors are similar to SIFT.
- It has also been shown the importance to have features uniformly distributed on the image, instead of having many but concentrated ones in certain areas. If some areas completely lack of feature points, this area has no reference points so it will not be properly reconstructed.
- Validation results have been given by using the following databases: CMUface [Paladini et al., 2010], Flag [White et al., 2007] and Rendered flag sequence [Garg et al., 2011] with a special focus on the last one, as it is a full rendered sequence with a full dense ground truth behind against the results could be compared with.
- In order to reduce the inherent ambiguities of the NRSfM technique and make the tracking thread compatible with it, a some information was introduced in the estimator in the form of time and shape priors. These help to reduce the flickering, which is noticeable on the 3D reconstruction error, and also stabilizes the estimated trajectory.
- Most of the state-of-the-art proposals work with projection orthographic. Our algorithm handles perspective projection, so it is able of reducing the ambiguities that the orthographic projection approaches have to face regarding the problem of depth estimation.
- The PTAM re-localization algorithm is analyzed for non-rigid sequences even though the active search [Williams et al., 2007] was developed for rigid objects. It has been demonstrated that, the best recovery is the based in pose and shape update, although for high quality tracking features a pose update is enough in case it is needed.



- In order to obtain good results, a good initialization of the tracking thread is fundamental, otherwise, the algorithm will do its best effort until it gets lost. This is the main reason why a mapping thread working in parallel that continually optimizes the map will give a good initialization to the tracking thread and will do the whole system works properly.

Hereafter, the main conclusions and contributions regarding the sequential map estimation and update of the mapping thread are summarized:

- The substitution of the Sparse Bundle Adjustment (SBA) technique based on the simplified version of the sparse Levenberg-Marquard (LM) in [Hartley and Zisserman, 2004] and used for PTAM, by a batch NRSfM algorithm, running in a sequential way with a set of keyframes has been validated, getting similar results as the original approach that take that all the frames, but in our case we reach a better time performance.
- The chosen NRSfM core algorithm, Metric Projection (MP), is able to reject the missing data and outliers natively during the optimization process. The obtained performance can be comparable with the baseline. Even though inputs are not ideal, the updated bases have quality enough to be used by the tracking process to maintain the whole system stable.
- With the presented approach, the mapping is able to adapt to changes in the deformation model for deformations that have not seen before, being the new bases updated and sending to the tracking thread.

As a main conclusion of the system as a whole, both subsystems working together make the whole NRSfM system more stable, scalable and adaptable to changes. In this way, we present an sparse NRSfM proposal, based on PTAM method but adapted to not-rigid objects, which supposes a good trade-off between reconstruction error and processing time, being compatible for real-time applications.

## 5.2 Future Works

This section depicts future works derived from the conclusions and contributions of this Thesis. As in the previous section, a similar division between tracking and mapping is carried out.

First, we review the main works to be done for the tracking thread:

- Other binary type of features such as LDB [Yang and Cheng, 2013], or any other existing ones, apart from the tested, should also be tested to check if the performance is increased without penalty in the rest of the parameters.

- Extra optimizations on the tracking thread code for real images are needed, redundancy checking and other careful procedures must be carried out to reduce the amount of necessary processing and get a frame rate of 30 and even 60 fps for VGA at the same time that thousands of points are handled on the map with the minimum impact on estimation quality.
- Performance tests with dense deformation bases should be carried out.
- DaLI features [Simo-Sierra et al., 2015] can be integrated if extreme deformations are foreseen to be present, although its impact on real-time performance should also be considered.

Regarding the mapping we propose the following future works:

- The full automatic initialization is an important problem that must be tackled. The initial rigid shape, camera pose and initial set of “accurate” deformation bases must be computed by an initialization process in an automatic way.

The initial rigid model can be estimated by a rigid factorization given by [Marques and Costeira, 2008], as already was successfully used by [Paladini et al., 2010, Agudo and Moreno-Noguer, 2015a]. A stereo pair initialization with deformable tracks gives very frequent losses, as many of the tracks do not follow the same epipolar line, so many of times those tracks are rejected. Even though [Marques and Costeira, 2008] works with orthographic projection, an upgrade could be performed to obtain the equivalent perspective pose. Once the rigid shape is estimated, the deformation shapes could be estimated from the given tracks using exhaustive evaluation of a batch NRSfM approach.

- Factorization-based algorithms are not fully adapted to outliers rejection. This problem could be avoided on the tracking thread not evaluating the affected points, although, for factorization algorithms all the points must be available. Therefore, there should be some robust factorization on the mapping thread to filter or reconstruct the scene avoiding the missing data or outliers.
- In order to speed up the processing of batch algorithms, following a similar approach to local SBA, we propose to go deeper into the sliding window scheme to update the bases.
- To implement real applications with our method, further code optimization will be necessary, cleaning up interdependencies, adaptations, libraries, redundancies, etc, as a previous step for an integration on an embedded platform such as NDK (Native Development Kit, for Android platforms).
- The mapping must be able to grow the area is being reconstructed, as it does PTAM when the camera explores a new area. As we use a keyframe-based approach, not

being limited to the reference frame, a strategy as the original PTAM can be followed. This leads to an approach similar to [Fayad et al., 2010, Russell et al., 2011] but expanded to the camera poses, not only to the local patches inside the frame. A similar problem was already faced by PTAM when the keyframes have points in common before discovering new areas.

- The mapping must be able to control the growing of the number of estimated bases, refactoring them to a lower number when they overtake a certain threshold. After this refactoring process, the coefficients of each frame on the map should be recomputed to be adapted to the new set of bases.



# Bibliography

- [Aanaes and Kahl, 2002] H. Aanaes and F. Kahl, 2002, ‘Estimation of deformable structure and motion’’. In *In Workshop on Vision and Modelling of Dynamic Scenes, ECCV*.
- [Agarwal et al., 2009] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz and R. Szeliski, 2009, ‘Building rome in a day’. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, 72–79.
- [Agudo et al., 2012a] A. Agudo, B. Calvo and J. M. M. Montiel, 2012a, ‘3D reconstruction of non-rigid surfaces in real-time using wedge elements’. In *Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment (ECCVW)*, 113–122.
- [Agudo et al., 2012b] A. Agudo, B. Calvo and J. M. M. Montiel, 2012b, ‘Finite element based sequential bayesian non-rigid structure from motion’. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 1418–1425.
- [Agudo et al., 2014] A. Agudo, J. M. M. Montiel, L. Agapito and B. Calvo, 2014, ‘Online dense non-rigid 3D shape and camera motion recovery’. In *British Machine Vision Conference*.
- [Agudo and Moreno-Noguer, 2015a] A. Agudo and F. Moreno-Noguer, 2015a, ‘Learning shape, motion and elastic models in force space’. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 756–764.
- [Agudo and Moreno-Noguer, 2015b] A. Agudo and F. Moreno-Noguer, 2015b, ‘Simultaneous pose and non-rigid shape with particle dynamics’. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Agudo et al., 2016a] A. Agudo, F. Moreno-Noguer, B. Calvo and J. Montiel, 2016a, ‘Real-time 3d reconstruction of non-rigid shapes with a single moving camera’. *Computer Vision and Image Understanding*.
- [Agudo et al., 2016b] A. Agudo, F. Moreno-Noguer, B. Calvo and J. Montiel, 2016b, ‘Sequential non-rigid structure from motion using physical priors’. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38(5):979–994.

- [Akhter et al., 2009] I. Akhter, Y. Sheikh, S. Khan and T. Kanade, 2009, ‘Nonrigid structure from motion in trajectory space’. In *Advances in neural information processing systems*, 41–48.
- [Alcantarilla et al., 2012] P. F. Alcantarilla, A. Bartoli and A. J. Davison, 2012, ‘Kaze features’. In *European Conference on Computer Vision*, 214–227, Springer.
- [Alcantarilla et al., 2013] P. F. Alcantarilla, J. Nuevo and A. Bartoli, 2013, ‘Fast explicit diffusion for accelerated features in nonlinear scale spaces’. In *British Machine Vision Conference (BMVC)*.
- [Bartoli and Collins, 2013] A. Bartoli and T. Collins, 2013, ‘Template-based isometric deformable 3d reconstruction with sampling-based focal length self-calibration’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Bartoli et al., 2015] A. Bartoli, Y. Gérard, F. Chadebecq, T. Collins and D. Pizarro, 2015, ‘Shape-from-template’. *IEEE transactions on pattern analysis and machine intelligence*, 37(10):2099–2118.
- [Bay et al., 2006] H. Bay, T. Tuytelaars and L. Van Gool, 2006, ‘Surf: Speeded up robust features’. In *European conference on computer vision*, 404–417, Springer.
- [Benhimane and Malis, 2007] S. Benhimane and E. Malis, 2007, ‘Homography-based 2d visual tracking and servoing’. *International Journal of Robotic Research*, 26(7):661–676.
- [Bernard et al., 2016] F. Bernard, P. Gemmar, F. Hertel, J. M. Gonçalves and J. Thunberg, 2016, ‘Linear shape deformation models with local support using graph-based structured matrix factorisation’. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, abs/1510.08291.
- [Booth et al., 2016] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah and D. Dunaway, 2016, ‘A 3d morphable model learnt from 10,000 faces’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Bregler et al., 1999] C. Bregler, A. Hertzmann and H. Biermann, 1999, ‘Recovering non-rigid 3D shape from image streams’. In *IEEE Conference on Computer Vision and Pattern Recognition*, 690–696.
- [Bronte et al., 2014] S. Bronte, M. Paladini, L. M. Bergasa, L. Agapito and R. Arroyo, 2014, ‘Real-time sequential model-based non-rigid SFM’. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014*, 1026–1031.
- [Brunet et al., 2010] F. Brunet, R. Hartley, A. Bartoli, N. Navab and R. Malgouyres, 2010, ‘Monocular template-based reconstruction of smooth and inextensible surfaces’. In R. Kimmel, R. Klette and A. Sugimoto, eds., *Proceedings of the Tenth*

- Asian Conference on Computer Vision (ACCV 2010)*, vol. 6494 of *Lecture Notes in Computer Science*, 52–66, Queenstown (New Zealand).
- [CGAL, 2007] CGAL, 2007, ‘CGAL, Computational Geometry Algorithms Library’. [Http://www.cgal.org](http://www.cgal.org).
- [Chhatkuli et al., 2014a] A. Chhatkuli, D. Pizarro and A. Bartoli, 2014a, ‘Non-rigid shape-from-motion for isometric surfaces using infinitesimal planarity’. In *British Machine Vision Conference, BMVC 2014, Nottingham, UK, September 1-5, 2014*.
- [Chhatkuli et al., 2014b] A. Chhatkuli, D. Pizarro and A. Bartoli, 2014b, ‘Stable template-based isometric 3d reconstruction in all imaging conditions by linear least-squares’. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, 708–715.
- [Chhatkuli et al., 2016] A. Chhatkuli, D. Pizarro, T. Collins and A. Bartoli, 2016, ‘Inextensible non-rigid shape-from-motion by second-order cone programming’. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 1719–1727.
- [Cohen et al., 2015] A. Cohen, T. Sattler and M. Pollefeys, 2015, ‘Merging the unmatched: Stitching visually disconnected sfm models’.
- [Collins and Bartoli, 2015] T. Collins and A. Bartoli, 2015, ‘Realtime shape-from-template: System and applications’. In *2015 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2015, Fukuoka, Japan, September 29 - Oct. 3, 2015*, 116–119.
- [Cootes et al., 1998] T. Cootes, G. Edwards and C. Taylor, 1998, ‘Active appearance models’. In *IEEE European Conference on Computer Vision*, 484–498.
- [Cui and Tan, 2015] Z. Cui and P. Tan, 2015, ‘Global structure-from-motion by similarity averaging’. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 864–872.
- [Curless and Levoy, 1996] B. Curless and M. Levoy, 1996, ‘A volumetric method for building complex models from range images’. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, New Orleans, LA, USA, August 4-9, 1996*, 303–312.
- [Dai et al., 2012] Y. Dai, H. Li and M. He, 2012, ‘A simple prior-free method for non-rigid structure-from-motion factorization’. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, 2018–2025.
- [Davison et al., 2007] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, 2007, ‘Monoslam: Real-time single camera slam’. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067.

- [Del Bue et al., 2006] A. Del Bue, X. Lladó and L. Agapito, 2006, ‘Non-rigid metric shape and motion recovery from uncalibrated images using priors’. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, 17-22 June 2006, New York, NY, USA, 1191–1198.
- [Dellaert, 2002] F. Dellaert, 2002, ‘The expectation maximization algorithm’. Tech. rep., College of Computing, Georgia Institute of Technology.
- [Dellaert and Kaess, 2006] F. Dellaert and M. Kaess, 2006, ‘Square root sam: Simultaneous localization and mapping via square root information smoothing’. *Int. J. Rob. Res.*, 25(12):1181–1203.
- [Eriksson et al., 2016] A. Eriksson, J. Bastian, T.-J. Chin and M. Isaksson, 2016, ‘A consensus-based framework for distributed bundle adjustment’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Fayad et al., 2010] J. Fayad, L. Agapito and A. Del Bue, 2010, ‘Piecewise quadratic reconstruction of non-rigid surfaces from monocular sequences’. In *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*, 297–310.
- [Fua and Salzmann, 2011] P. Fua and M. Salzmann, 2011, ‘Linear local models for monocular reconstruction of deformable surfaces’. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(undefined):931–944.
- [Gallardo et al., 2016] M. Gallardo, T. Collins and A. Bartoli, 2016, ‘Using shading and a 3d template to reconstruct complex surface deformations’. *BMVC 2016*.
- [Garg et al., 2013] R. Garg, A. Roussos and L. Agapito, 2013, ‘Dense variational reconstruction of non-rigid surfaces from monocular video’. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 1272–1279.
- [Garg et al., 2011] R. Garg, A. Roussos and L. de Agapito, 2011, ‘Robust trajectory-space tv-l1 optical flow for non-rigid sequences’. In *EMMCVPR*, vol. 6819, 300–314.
- [Gong et al., 2015] Y. Gong, D. Meng and E. J. Seibel, 2015, ‘Bound constrained bundle adjustment for reliable 3d reconstruction’. *Optics express*, 23(8):10771–10785.
- [Gotardo and Martinez, 2011] P. F. Gotardo and A. M. Martinez, 2011, ‘Non-rigid structure from motion with complementary rank-3 spaces’. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3065–3072.
- [Gower and Dijksterhuis, 2004] J. C. Gower and G. B. Dijksterhuis, 2004, *Procrustes Problems*.
- [Haouchine et al., 2014] N. Haouchine, J. Dequidt, M. Berger and S. Cotin, 2014, ‘Single view augmentation of 3d elastic objects’. In *IEEE International Symposium on Mixed*



- and Augmented Reality, ISMAR 2014, Munich, Germany, September 10-12, 2014*, 229–236.
- [Hartley and Vidal, 2008] R. I. Hartley and R. Vidal, 2008, ‘Perspective nonrigid shape and motion recovery’. In *Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part I*, 276–289.
- [Hartley and Zisserman, 2004] R. I. Hartley and A. Zisserman, 2004, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edn.
- [Itseez, 2015] Itseez, 2015, ‘Open source computer vision library’. <https://github.com/itseez/opencv>.
- [Joseph Tan et al., 2014] D. Joseph Tan, S. Holzer, N. Navab and S. Ilic, 2014, ‘Deformable template tracking in 1ms’. In *Proceedings of the British Machine Vision Conference*, BMVA Press.
- [Klein, 2006] G. Klein, 2006, ‘Visual tracking for augmented reality’. Ph.D. thesis, University of Cambridge, Cambridge, United Kingdom.
- [Klein and Murray, 2007] G. Klein and D. Murray, 2007, ‘Parallel tracking and mapping for small AR workspaces’. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 1–10.
- [Klein and Murray, 2008] G. Klein and D. Murray, 2008, ‘Improving the agility of keyframe-based SLAM’. In *Proc. 10th European Conference on Computer Vision (ECCV’08)*, 802–815, Marseille.
- [Klein and Murray, 2009] G. Klein and D. Murray, 2009, ‘Parallel tracking and mapping on a camera phone’. In *Proc. Eighth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR’09)*, Orlando.
- [Kong and Lucey, 2016] C. Kong and S. Lucey, 2016, ‘Prior-less compressible structure from motion’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Lee et al., 2013] M. Lee, J. Cho, C. Choi and S. Oh, 2013, ‘Procrustean normal distribution for non-rigid structure from motion’. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, 1280–1287.
- [Lee et al., 2016] M. Lee, J. Cho and S. Oh, 2016, ‘Consensus of non-rigid reconstructions’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Leizea et al., 2014] I. Leizea, H. Álvarez, I. Aguinaga and D. Borro, 2014, ‘Real-time deformation, registration and tracking of solids based on physical simulation’. In

- IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2014, Munich, Germany, September 10-12, 2014*, 165–170.
- [Leizea et al., 2015] I. Leizea, H. Álvarez and D. Borro, 2015, ‘Real time non-rigid 3d surface tracking using particle filter’. *Computer Vision and Image Understanding*, 133:51 – 65.
- [Leutenegger et al., 2011] S. Leutenegger, M. Chli and R. Y. Siegwart, 2011, ‘Brisk: Binary robust invariant scalable keypoints’. In *2011 International conference on computer vision*, 2548–2555, IEEE.
- [Liu-Yin et al., 2016] Q. Liu-Yin, R. Yu, A. Fitzgibbon, L. Agapito and C. Russell, 2016, ‘Better together: Joint reasoning for non-rigid 3d reconstruction with specularities and shading’. *BMVC 2016*.
- [Lladó et al., 2005] X. Lladó, A. Del Bue and L. Agapito, 2005, ‘Non-rigid 3d factorization for projective reconstruction’. In *Proceedings of the British Machine Vision Conference 2005, Oxford, UK, September 2005*.
- [Lowe, 2004] D. G. Lowe, 2004, ‘Distinctive image features from scale-invariant keypoints’. *International journal of computer vision*, 60(2):91–110.
- [Magenat et al., 2015] S. Magneat, D. T. Ngo, F. ZÄ¼nd, M. Ryffel, G. Noris, G. Roethlin, A. Marra, M. Nitti, P. Fua, M. H. Gross and R. W. Sumner, 2015, ‘Live texturing of augmented reality characters from colored drawings’. *IEEE Trans. Vis. Comput. Graph.*, 21(11):1201–1210.
- [Maier-Hein et al., 2014] L. Maier-Hein, A. Groch, A. Bartoli, S. Bodenstedt, G. Boissonnat, P.-L. Chang, N. Clancy, D. S. Elson, S. Haase, E. Heim et al., 2014, ‘Comparative validation of single-shot optical techniques for laparoscopic 3-d surface reconstruction’. *IEEE transactions on medical imaging*, 33(10):1913–1930.
- [Marques and Costeira, 2008] M. Marques and J. Costeira, 2008, ‘Optimal shape from motion estimation with missing and degenerate data’. In *2008 IEEE Workshop on Motion and video Computing*, 1–6.
- [Moreno-Noguer and Fua, 2013] F. Moreno-Noguer and P. Fua, 2013, ‘Stochastic exploration of ambiguities for nonrigid shape recovery’. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(2):463–475.
- [Moreno-Noguer and Porta, 2011] F. Moreno-Noguer and J. M. Porta, 2011, ‘Probabilistic simultaneous pose and non-rigid shape recovery’. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1289–1296.
- [Moreno-Noguer et al., 2009] F. Moreno-Noguer, M. Salzmann, V. Lepetit and P. Fua, 2009, ‘Capturing 3d stretchable surfaces from single images in closed form’. In *2009*

- IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, 20-25 June 2009, Miami, Florida, USA, 1842–1849.
- [Muñoz et al., 2009] E. Muñoz, J. M. Buenaposada and L. Baumela, 2009, ‘A direct approach for efficiently tracking with 3D morphable models’. In *International Conference on Computer Vision*, 1615–1622.
- [Newcombe and Davison, 2010] R. A. Newcombe and A. J. Davison, 2010, ‘Live dense reconstruction with a single moving camera’. In *IEEE Conference on Computer Vision and pattern Recognition*.
- [Newcombe et al., 2015] R. A. Newcombe, D. Fox and S. M. Seitz, 2015, ‘Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Newcombe et al., 2011a] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, K. Pushmeet, J. Shotton, S. Hodges and A. Fitzgibbon, 2011a, ‘Kinectfusion: Real-time dense surface mapping and tracking’. In *IEEE ISMAR*, IEEE.
- [Newcombe et al., 2011b] R. A. Newcombe, S. J. Lovegrove and A. J. Davison, 2011b, ‘DTAM: Dense tracking and mapping in real-time’. In *International Conference on Computer Vision*, 2320–2327.
- [Ngo et al., 2016] D. T. Ngo, J. Östlund and P. Fua, 2016, ‘Template-based monocular 3d shape recovery using laplacian meshes’. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(1):172–187.
- [Ngo et al., 2015] D. T. Ngo, S. Park, A. Jorstad, A. Crivellaro, C. Yoo and P. Fua, 2015, ‘Handling occlusions and sparse textures in a deformable surface tracking framework’. *CoRR*, abs/1503.03429.
- [Ondruska et al., 2015] P. Ondruska, P. Kohli and S. Izadi, 2015, ‘Mobilefusion: Real-time volumetric surface reconstruction and dense tracking on mobile phones’. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Fukuoka, Japan.
- [Paladini, 2011] M. Paladini, 2011, ‘Deformable and articulated 3d reconstruction from monocular video sequences’. Ph.D. thesis, Queen Mary, University of London, London, United Kingdom.
- [Paladini et al., 2010] M. Paladini, A. Bartoli and L. Agapito, 2010, ‘Sequential non-rigid structure-from-motion with the 3D-implicit low-rank shape model’. In *Proceedings of the 11th European Conference on Computer vision*, 15–28.
- [Paladini et al., 2009] M. Paladini, A. Del Bue, M. Stosic, M. Dodig, J. Xavier and L. Agapito, 2009, ‘Factorization for non-rigid and articulated structure using met-

- ric projections’. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2898–2905.
- [Paladini et al., 2012] M. Paladini, A. Del Bue, J. M. F. Xavier, L. Agapito, M. Stosic and M. Dodig, 2012, ‘Optimal metric projections for deformable and articulated structure-from-motion’. *International Journal of Computer Vision*, 96(2):252–276.
- [Pan et al., 2009] Q. Pan, G. Reitmayr and T. W. Drummond, 2009, ‘Interactive model reconstruction with user guidance’. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, 209–210, IEEE.
- [Parashar et al., 2016] S. Parashar, D. Pizarro and A. Bartoli, 2016, ‘Isometric non-rigid shape-from-motion in linear time’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Parashar et al., 2015] S. Parashar, D. Pizarro, A. Bartoli and T. Collins, 2015, ‘As-rigid-as-possible volumetric shape-from-template’. In *The IEEE International Conference on Computer Vision (ICCV)*.
- [Paulus et al., 2015] C. J. Paulus, N. Haouchine, D. Cazier and S. Cotin, 2015, ‘Augmented reality during cutting and tearing of deformable objects’. In *2015 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2015, Fukuoka, Japan, September 29 - Oct. 3, 2015*, 54–59.
- [Pizarro and Bartoli, 2012] D. Pizarro and A. Bartoli, 2012, ‘Feature-based deformable surface detection with self-occlusion reasoning’. *International Journal of Computer Vision*, 97(1):54–70.
- [Pradeep et al., 2013] V. Pradeep, C. Rhemann, S. Izadi, C. Zach, M. Bleyer and S. Bathiche, 2013, ‘Monofusion: Real-time 3d reconstruction of small scenes with a single web camera’. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, 83–88.
- [Ranftl et al., 2016] R. Ranftl, V. Vineet, Q. Chen and V. Koltun, 2016, ‘Dense monocular depth estimation in complex dynamic scenes’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Rosten and Drummond, 2006] E. Rosten and T. Drummond, 2006, ‘Machine learning for high-speed corner detection’. In *European Conference on Computer Vision*, vol. 1, 430–443.
- [Rublee et al., 2011] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, 2011, ‘Orb: An efficient alternative to sift or surf’. In *2011 International conference on computer vision*, 2564–2571, IEEE.
- [Russell et al., 2011] C. Russell, J. Fayad and L. Agapito, 2011, ‘Energy based multiple model fitting for non-rigid structure from motion’. In *The 24th IEEE Conference*

- on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, 3009–3016.
- [Russell et al., 2014] C. Russell, R. Yu and L. Agapito, 2014, ‘Video pop-up: Monocular 3d reconstruction of dynamic scenes’. In D. Fleet, T. Pajdla, B. Schiele and T. Tuytelaars, eds., *Computer Vision – ECCV 2014*, vol. 8695 of *Lecture Notes in Computer Science*, 583–598, Springer.
- [Rusu and Cousins, 2011] R. B. Rusu and S. Cousins, 2011, ‘3d is here: Point cloud library (pcl)’. In *International Conference on Robotics and Automation*, Shanghai, China.
- [Salzmann et al., 2008] M. Salzmann, F. Moreno-Noguer, V. Lepetit and P. Fua, 2008, ‘Closed-form solution to non-rigid 3d surface registration’. In *Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV*, 581–594.
- [Sanchez-Riera et al., 2010] J. Sanchez-Riera, J. Östlund, P. Fua and F. Moreno-Noguer, 2010, ‘Simultaneous pose, correspondence and non-rigid shape’. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, 1189–1196.
- [Sarbolandi et al., 2015] H. Sarbolandi, D. Lefloch and A. Kolb, 2015, ‘Kinect range sensing: Structured-light versus time-of-flight kinect’. *Computer Vision and Image Understanding*, 139:1 – 20.
- [Schonberger and Frahm, 2016] J. L. Schonberger and J.-M. Frahm, 2016, ‘Structure-from-motion revisited’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Schöps et al., 2014] T. Schöps, J. Engel and D. Cremers, 2014, ‘Semi-dense visual odometry for AR on a smartphone’. In *IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2014, Munich, Germany, September 10-12, 2014*, 145–150.
- [Simo-Sierra et al., 2015] E. Simo-Sierra, C. Torras and F. Moreno-Noguer, 2015, ‘DaLI: Deformation and light invariant descriptor’.
- [Steinbruecker et al., 2014] F. Steinbruecker, J. Sturm and D. Cremers, 2014, ‘Volumetric 3d mapping in real-time on a cpu’. Hongkong, China.
- [Stewenius et al., 2006] H. Stewenius, C. Engels and D. Nistér, 2006, ‘Recent developments on direct relative orientation’. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):284–294.
- [Stühmer et al., 2010] J. Stühmer, S. Gumhold and D. Cremers, 2010, ‘Real-time dense geometry from a handheld camera’. In *DAGM-Symposium*, 11–20.

- [Sturm and Triggs, 1996] P. F. Sturm and B. Triggs, 1996, ‘A factorization based algorithm for multi-image projective structure and motion’. In *Proceedings of the 4th European Conference on Computer Vision-Volume II - Volume II*, ECCV ’96, 709–720, Springer-Verlag, London, UK, UK.
- [Tan et al., 2013] W. Tan, H. Liu, Z. Dong, G. Zhang and H. Bao, 2013, ‘Robust monocular SLAM in dynamic environments’. In *IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2013, Adelaide, Australia, October 1-4, 2013*, 209–218.
- [Tang and Hung, 2002] W. K. Tang and Y. S. Hung, 2002, *A Factorization-Based Method for Projective Reconstruction with Minimization of 2-D Reprojection Errors*, 387–394. Springer Berlin Heidelberg, Berlin, Heidelberg, URL [http://dx.doi.org/10.1007/3-540-45783-6\\_47](http://dx.doi.org/10.1007/3-540-45783-6_47).
- [Tomasi and Kanade, 1992] C. Tomasi and T. Kanade, 1992, ‘Shape and motion from image streams under orthography: a factorization method’. *International Journal of Computer Vision*, 9(2):137–154.
- [Torresani et al., 2003] L. Torresani, A. Hertzmann and C. Bregler, 2003, ‘Learning non-rigid 3d shape from 2d motion’. In *Advances in Neural Information Processing Systems 16*, 1555–1562.
- [Torresani et al., 2008] L. Torresani, A. Hertzmann and C. Bregler, 2008, ‘Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors’. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, 878–892.
- [Triggs et al., 1999] B. Triggs, P. F. McLauchlan, R. I. Hartley and A. W. Fitzgibbon, 1999, ‘Bundle adjustment - A modern synthesis’. In *International workshop on vision algorithms*, 298–372, Springer.
- [Tukey, 1960] J. W. Tukey, 1960, ‘A survey of sampling from contaminated distributions’. *Contributions to probability and statistics*, 2:448–485.
- [Varol et al., 2009] A. Varol, M. Salzmann, E. Tola and P. Fua, 2009, ‘Template-free monocular reconstruction of deformable surfaces’. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, 1811–1818.
- [Vicente and Agapito, 2012] S. Vicente and L. Agapito, 2012, ‘Soft inextensibility constraints for template-free non-rigid reconstruction’. In *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part III*, 426–440.
- [Wang et al., 2016] G. Wang, Z. Wang, Y. Chen, Q. Zhou and W. Zhao, 2016, ‘Context-aware gaussian fields for non-rigid point set registration’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- [Wang et al., 2015] S. Wang, S. Fidler and R. Urtasun, 2015, ‘Lost shopping! monocular localization in large indoor spaces’. In *The IEEE International Conference on Computer Vision (ICCV)*.
- [White et al., 2007] R. White, K. Crane and D. Forsyth, 2007, ‘Capturing and animating occluded cloth’. In *ACM Transactions on Graphics (SIGGRAPH)*, 34.
- [Williams et al., 2007] B. Williams, G. Klein and I. Reid, 2007, ‘Real-time SLAM relocalisation’. In *Proc. International Conference on Computer Vision*.
- [Yang and Cheng, 2013] X. Yang and K. T. T. Cheng, 2013, ‘Local difference binary for ultrafast and distinctive feature description’. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):188–194.
- [Yu et al., 2015] R. Yu, C. Russell, N. D. F. Campbell and L. Agapito, 2015, ‘Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video’. In *The IEEE International Conference on Computer Vision (ICCV)*.
- [Zhou et al., 2016] Y. Zhou, E. Antonakos, J. Alabort-i Medina, A. Roussos and S. Zafeiriou, 2016, ‘Estimating correspondences of deformable objects "in-the-wild"’. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.





## Appendix A

# Translation-size ambiguity study

For the rigid case this ambiguity can appear because as it is said, the reconstructions are given up to scale. Actually, there is no certain way to assure the correct dimension of an object unless a certain reference is given.

Once the size is fixed, for the rigid case, the ambiguity does no longer exists, as there is no more factors that influence the size of the reconstruction along the time.

However, on the case of the non rigid structure from motion, even though we could know the object size at the beginning and assuming it has rigid size, if no other prior or underlying model is assumed that constrain the deformations, the estimations could lead to incorrect estimations of the translation vector together with the size of the non-rigid component.

It is actually an effect of the distributive property applied to the shape model and the projection equation for multiple models.

In [Aanaes and Kahl, 2002] an indication of the metric projection matrix is depicted. As a result of a factorization, any metric projection matrix  $G$  could be put in the middle, so that the following decomposition can be done:

$$W = UV = UGG^{-1}V \tag{A.1}$$

The final result in the reconstruction would be the same. This is valid for orthographic projection, although it gives an idea of how ambiguous a reconstruction could be.

Many recent works, like [Paladini et al., 2009, Dai et al., 2012, Kong and Lucey, 2016] have concentrated on computing this metric projection matrix by imposing several constraints on the optimization process, so that it could be unique, the rank of the solution would be minimum, etc.

For the perspective projection the problem still holds, as the orthographic projection does not take into account the camera-object depth, and the same restrictions can not be applied, as there is a non-linear dependence on the depth component.

Therefore, for a perspective projection there is an inherent ambiguity between the size of the object and the depth that an estimator could obtain out of even if only non-rigid tracks are provided. The demonstration is as follows:

The projection equation for the rigid case is, in homogeneous coordinates:

$$U = K [R|T] \begin{pmatrix} S \\ 1 \end{pmatrix} \quad (\text{A.2})$$

When this equation is extended to the non-rigid domain, the shape term is extended so as to comply with the linear deformation model, such as:

$$S = S_R + S_{NR} \quad (\text{A.3})$$

For rigid reconstructions, the rigid term  $S_R$  is fixed once the scale is fixed and  $S_{NR}$  is zero for all the frames.

For the non-rigid case, the term  $S_{NR}$  is not 0 and, if no limits are imposed,

$$S_{NR} = LB \quad (\text{A.4})$$

A size change on the object could be interpreted as a change on the scale of the coefficients. Therefore, to represent this in the previous equations, a scale  $a$  could be introduced in the coefficients,

$$S'_{NR} = aLB \quad (\text{A.5})$$

Now we demonstrate the same projections could be obtained with  $S_{NR}$  and  $S'_{NR}$ , but some other parts on the projection equation must change to preserve the final values.

The projection with the original  $S_{NR}$  is:

$$U = K [R|T] \begin{pmatrix} (S_R + S_{NR}) \\ 1 \end{pmatrix} = KRS_R + KRS_{NR} + KT \quad (\text{A.6})$$

and the one with  $S'_{NR} = aS_{NR}$ :

$$U = K [R|T'] (S_R + S'_{NR}) = KRS_R + KRS'_{NR} + KT' \quad (\text{A.7})$$

$$U = KRS_R + KR aLB + KT' \quad (\text{A.8})$$

As  $U$  is the same in both equations,

$$aKRLB + KT' = KRLB + KT \quad (\text{A.9})$$

multiplying by the left by  $K^{-1}$

$$aRLB + T' = RLB + T \quad (\text{A.10})$$

$$T' = T + (1 - a)RLB = T + (1 - a)RS_{NR} \quad (\text{A.11})$$

If the estimation of the coefficients starts failing in its size, translation could start failing, compensating the error by the factor indicated previously.

In order to avoid this effect other information apart from just *reprojection* must be included on the estimation, as it is not enough to disambiguate among possible solutions, as already was seen on the priors section on the related works section.



## Appendix B

### Prior derivation

This appendix shows the derivation of temporal and spatial smoothness priors in our Model-based deformable reconstruction method described in Chapter 3. These priors are an important tool to better constrain the solution.

#### B.1 Temporal smoothness

This prior is usually imposed in the literature to reduce flickering between frames. It is based on penalizing the difference between the reconstruction solution at current frame with respect to the reconstruction solution in the last frame.

It can be applied only to the 3D shape as follows:

$$E_{temp} = \sum_{i=1}^P \|S_i(f) - S_i(f-1)\|^2 \quad (\text{B.1})$$

Eq. (B.1) is rewritten as:

$$E_{temp} = \sum_{i=1}^P \|B_i L^T(f) - S_i(f-1)\|^2 \quad (\text{B.2})$$

where  $B_i$  is the  $3 \times K$  portion of the basis that corresponds to the  $i$ -th point and  $L = (l_1 \cdots l_K)$  are the shape deformation coefficients. Last frame's shape  $S_i(f-1)$  is considered here known and not dependent on  $L$ .

By rewriting Eq. (B.2) into a Linear Least Squares (LLS) system we have:

$$E_{temp} = \|A_{temp}L - B_{temp}\|^2 \quad (\text{B.3})$$

where:

$$A_{temp} = \begin{pmatrix} B_1^\top & \cdots & B_P^\top \end{pmatrix}^\top \quad B_{temp} = \begin{pmatrix} S_1^\top(f-1) & \cdots & S_P^\top(f-1) \end{pmatrix}^\top \quad (\text{B.4})$$

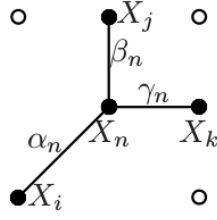


Figure B.1: Shape smoothness graphical representation

Eq. (B.4) can be directly included in our reconstruction method as it is quadratic in the shape deformation parameters  $L$ . This means that it becomes a linear equation in the E-step described in 3, that is solved with LSS.

## B.2 Spatial smoothness

Spatial smoothness is based on forcing neighboring relationships between points in the mesh to be coherent to the mesh at rest. Coherency means here that a point in the shape is estimated with same coordinates of 3 of his neighbours.

Fig. B.1 shows 4 points that belong to the surface, where  $X_n$  is the point of interest and  $(X_i, X_j, X_k)$  are three non-collinear neighbours.  $X_n$  can be thus represented by a set of coordinates  $(\alpha_n, \beta_n, \gamma_n)$  that are obtained as the solution of the following linear system:

$$\begin{pmatrix} X_{i,x} & X_{j,x} & X_{k,x} \\ X_{i,y} & X_{j,y} & X_{k,y} \\ X_{i,z} & X_{j,z} & X_{k,z} \end{pmatrix} \begin{pmatrix} \alpha_n \\ \beta_n \\ \gamma_n \end{pmatrix} = \begin{pmatrix} X_{n,x} \\ X_{n,y} \\ X_{n,z} \end{pmatrix} \quad (\text{B.5})$$

Once each point of the mesh is defined with respect to its neighbours in a reference shape (for instance the first frame), the shape prior could be defined as:

$$E_{shape} = \sum_{n=1}^P \sum_{i,j,k \in \mathfrak{N}(X_n)} \|X_n - (\alpha_n X_i + \beta_n X_j + \gamma_n X_k)\|^2 \quad (\text{B.6})$$

where  $\mathfrak{N}(X_n)$  means the neighborhood of the point  $X_n$ . By substitution of the deformation parameters in Eq. (B.6) it becomes the following homogeneous quadratic equation:

$$E_{shape} = \|A_{shape} L\|^2 \quad (\text{B.7})$$

where

$$A_{shape} = \begin{pmatrix} B'_1 \\ \vdots \\ B'_P \end{pmatrix} \quad (\text{B.8})$$

and

$$B'_n = B_n - \alpha_n B_i - \beta_n B_j - \gamma_n B_k \quad (\text{B.9})$$

### B.3 Isometry

Isometry prior, or at least the euclidean relaxation of inextensibility, would have been the perfect constraint for some of the test sequences, as the surfaces presented comply with the isometry conditions. The main problem is that the E-step of the presented minimization algorithm for the tracking solves the shape in closed form and it is based on LLS, whereas isometry or inextensibility would have required to introduce a fourth degree equation (due to the distances introduced), as it is the module of the difference of the distances

$$E_{iso} = \sum_{i=1}^P \sum_{j=1, j \neq i}^P \left\| d(X_i, X_j) - d(X_i^0, X_j^0) \right\|^2 \quad (\text{B.10})$$

$$\begin{aligned} \left\| \|X_i - X_j\|^2 - \|X_i^0 - X_j^0\|^2 \right\|^2 &= \left\| \|L(B_i - B_j)\|^2 - \|X_i^0 - X_j^0\|^2 \right\|^2 = \\ &= \left\| (B_i - B_j)^T L^T L (B_i - B_j) - (X_i^0 - X_j^0)^T (X_i^0 - X_j^0) \right\|^2 = \\ &= \left\| (B_i - B_j) (B_i - B_j)^T L^T L - (B_i - B_j) (X_i^0 - X_j^0)^T (X_i^0 - X_j^0) (B_i - B_j)^{-1} \right\|^2 \end{aligned} \quad (\text{B.11})$$

From the last equation it can be seen that the  $L$  term can not be solved linearly, as it is actually  $L^T L$ , which means that it can not be minimized via least squares as it introduces an equation of the form  $\|AL^2 - C\|^2 = 0$ .

### B.4 Aggregation of priors on the tracking solvers

As depicted in Chapter 3, the E-step requires solving the LLS system  $AL = B$  with  $L = \begin{pmatrix} l_1 & \dots & l_K \end{pmatrix}$  and

$$A = \begin{pmatrix} (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{11} & \dots & (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{1K} \\ (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{11} & \dots & (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{1K} \\ \vdots & \ddots & \vdots \\ (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{P1} & \dots & (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{PK} \\ (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{P1} & \dots & (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{PK} \end{pmatrix} \quad (\text{B.12})$$

$$B = \begin{pmatrix} f_u t_x - t_z \Delta u_1 \\ f_v t_y - t_z \Delta v_1 \\ \vdots \\ f_u t_x - t_z \Delta u_P \\ f_v t_y - t_z \Delta v_P \end{pmatrix} \quad (\text{B.13})$$

For each point, both the spatial and temporal priors are added on the corresponding matrix as additional rows in  $A$  and  $B$ . For instance, if the temporal smoothness prior is activated, the shape estimation is modified as follows:

$$A_{Shape} = \begin{pmatrix} (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{11} & \cdots & (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{1K} \\ (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{11} & \cdots & (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{1K} \\ \textcolor{red}{B'_{11,x}} & \cdots & \textcolor{red}{B'_{1K,x}} \\ \textcolor{red}{B'_{11,y}} & \cdots & \textcolor{red}{B'_{1K,y}} \\ \textcolor{red}{B'_{11,z}} & \cdots & \textcolor{red}{B'_{1K,z}} \\ \vdots & \ddots & \vdots \\ (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{P1} & \cdots & (\Delta u_P \vec{r}_z - f_u \vec{r}_x) B'_{PK} \\ (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{P1} & \cdots & (\Delta v_P \vec{r}_z - f_v \vec{r}_y) B'_{PK} \\ \textcolor{red}{B'_{P1,x}} & \cdots \cdots \cdots & \textcolor{red}{B'_{PK,x}} \\ \textcolor{red}{B'_{P1,y}} & \cdots \cdots \cdots & \textcolor{red}{B'_{PK,y}} \\ \textcolor{red}{B'_{P1,z}} & \cdots \cdots \cdots & \textcolor{red}{B'_{PK,z}} \end{pmatrix} \quad (\text{B.14})$$

$$B_{Shape} = \begin{pmatrix} f_u t_x - t_z \Delta u_1 \\ f_v t_y - t_z \Delta v_1 \\ \textcolor{red}{S_{1,x}(f-1)} \\ \textcolor{red}{S_{1,y}(f-1)} \\ \textcolor{red}{S_{1,z}(f-1)} \\ \vdots \\ f_u t_x - t_z \Delta u_P \\ f_v t_y - t_z \Delta v_P \\ \textcolor{red}{S_{P,x}(f-1)} \\ \textcolor{red}{S_{P,y}(f-1)} \\ \textcolor{red}{S_{P,z}(f-1)} \end{pmatrix} \quad (\text{B.15})$$

$$A_{Shape} L = B_{Shape} \quad (\text{B.16})$$

$$L = \begin{pmatrix} l_1 & \cdots & l_K \end{pmatrix}^T \quad (\text{B.17})$$

For simplicity, the product of the M-estimator weight and the prior weight is not indicated but it affects each of the point rows, depending on the measurement quality and the prior weight.



## Appendix C

# Closed form coefficient deduction

On chapter 3 the whole deduction of the closed form matrix was not fully specified. It is large enough to be included in a chapter, so it is moved to this Appendix.

In case the initial rigid shape can not be described as the sum of the non rigid components, an alternative version of the initially proposed method must be given to take out the influence of the rigid shape out of the projection equations.

It is important to mention that this estimation relies on a pose that is assumed to be fixed and correct. Otherwise it could yield wrong estimations on the coefficients estimated with this Weighted Least Squares (WLS) based algorithm.

### C.1 Rigid shape is a function of the bases

Given the linear basis deformation model:

$$S_{NR} = S_R + \sum_k L_k B_k \quad (C.1)$$

If possible, it could be considered that the  $S_R$  could be described as

$$S_R = \sum_k L_{Rk} B_k \quad (C.2)$$

Therefore, the whole shape could be estimated by  $L_k$ , instead of dividing the estimation in two parts, so the model would be:

$$S_{NR} = \sum_k L_k B_k \quad (C.3)$$

and then, given that the  $S_R$  is known in advance, the coefficients of the non-rigid part can be directly estimated by a simple subtraction.

$$L_{NR} = L - L_R \quad (\text{C.4})$$

In order to estimate the coefficients in closed form, we depart from the projection equation and its actual value.

$$U = K [R|T] \begin{pmatrix} S_{NR} \\ 1 \end{pmatrix} \quad (\text{C.5})$$

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \vec{r}_x & t_x \\ \vec{r}_y & t_y \\ \vec{r}_z & t_z \end{pmatrix} \begin{pmatrix} \sum_k L_k B_k \\ 1 \end{pmatrix} \quad (\text{C.6})$$

being the matrix  $R$  decomposed in its rows  $(\vec{r}_x, \vec{r}_y, \vec{r}_z)^T$  and  $T$  in its spacial components  $t_x, t_y$  and  $t_z$

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \vec{r}_x \sum_k L_k B_k + t_x \\ \vec{r}_y \sum_k L_k B_k + t_y \\ \vec{r}_z \sum_k L_k B_k + t_z \end{pmatrix} \quad (\text{C.7})$$

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u (\sum_k L_k \vec{r}_x B_k + t_x) + u_0 \lambda \\ f_v (\sum_k L_k \vec{r}_y B_k + t_y) + v_0 \lambda \\ \sum_k L_k \vec{r}_z B_k + t_z \end{pmatrix} \quad (\text{C.8})$$

leaving only the  $L_k$  at one side of the equation and the rest at the other side for each side of the terms, like  $\lambda u, \lambda v$ , as the only information that provides the last equation is that lambda is equals to an expression related to the coefficients and the rest of the variables.

Doing this to the  $u$  component:

$$\lambda u - \lambda u_0 = f_u \left( \sum_k L_k \vec{r}_x B_k + t_x \right) \quad (\text{C.9})$$

$$\sum_k L_k \vec{r}_z B_k (u - u_0) + t_z (u - u_0) = f_u \sum_k L_k \vec{r}_x B_k + f_u t_x \quad (\text{C.10})$$

$$\sum_k L_k \vec{r}_z B_k (u - u_0) - f_u \sum_k L_k \vec{r}_x B_k = f_u t_x - t_z (u - u_0) \quad (\text{C.11})$$

$$\sum_k L_k B_k (\vec{r}_z (u - u_0) - f_u \vec{r}_x) = f_u t_x - t_z (u - u_0) \quad (\text{C.12})$$

And applying the same to the  $v$  component:

$$\sum_k L_k B_k (\vec{r}_z (v - v_0) - f_v \vec{r}_y) = f_v t_y - t_z (v - v_0) \quad (\text{C.13})$$

In a similar manner to the way it was solved the Jacobian of the pose, a WLS is applied  $Ax = C$ , so each point will contribute to these matrix as follows:

$$A = \begin{pmatrix} (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{11} & \cdots & (\Delta u_1 \vec{r}_z - f_u \vec{r}_x) B'_{1k} \\ (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{11} & \cdots & (\Delta v_1 \vec{r}_z - f_v \vec{r}_y) B'_{1k} \\ \vdots & \ddots & \vdots \\ (\Delta u_p \vec{r}_z - f_u \vec{r}_x) B'_{p1} & \cdots & (\Delta u_p \vec{r}_z - f_u \vec{r}_x) B'_{pk} \\ (\Delta v_p \vec{r}_z - f_v \vec{r}_y) B'_{p1} & \cdots & (\Delta v_p \vec{r}_z - f_v \vec{r}_y) B'_{pk} \end{pmatrix} \quad (\text{C.14})$$

$$C = \begin{pmatrix} f_u t_x - t_z \Delta u_1 \\ f_v t_y - t_z \Delta v_1 \\ \vdots \\ f_u t_x - t_z \Delta u_p \\ f_v t_y - t_z \Delta v_p \end{pmatrix} \quad (\text{C.15})$$

being  $\Delta u_i = u_i - u_0$

## C.2 Taking into account rigid shape on the estimation

The model equation changes so as to include the rigid shape as a parameter

$$S_{NR} = S_R + \sum_k L_k B_k \quad (\text{C.16})$$

If  $S_R$  is not expressed as a function of the coefficients, as said in the previous approach, it is needed to be explicitly included in the equation.

Departing from the projection equations, things parameters change slightly, as follows:

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \vec{r}_x & t_x \\ \vec{r}_y & t_y \\ \vec{r}_z & t_z \end{pmatrix} \begin{pmatrix} S_R + \sum_k L_k B_k \\ 1 \end{pmatrix} \quad (\text{C.17})$$

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \vec{r}_x (S_R + \sum_k L_k B_k) + t_x \\ \vec{r}_y (S_R + \sum_k L_k B_k) + t_y \\ \vec{r}_z (S_R + \sum_k L_k B_k) + t_z \end{pmatrix} \quad (\text{C.18})$$

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} f_u (\vec{r}_x (S_R + \sum_k L_k B_k) + t_x) + u_0 \lambda \\ f_v (\vec{r}_y (S_R + \sum_k L_k B_k) + t_y) + v_0 \lambda \\ \vec{r}_z (S_R + \sum_k L_k B_k) + t_z \end{pmatrix} \quad (\text{C.19})$$

Doing the grouping of the coefficients with the  $u$  equation:

$$\lambda u - \lambda u_0 = f_u \left( \vec{r}_x \left( S_R + \sum_k L_k B_k \right) + t_x \right) \quad (\text{C.20})$$

$$\left( \vec{r}_z \left( S_R + \sum_k L_k B_k \right) + t_z \right) (u - u_0) = f_u \left( \vec{r}_x \left( S_R + \sum_k L_k B_k \right) + t_x \right) \quad (\text{C.21})$$

$$\vec{r}_z S_R (u - u_0) + (u - u_0) \vec{r}_z \sum_k L_k B_k + t_z (u - u_0) = f_u \vec{r}_x S_R + f_u \vec{r}_x \sum_k L_k B_k + f_u t_x \quad (\text{C.22})$$

$$(u - u_0) \vec{r}_z \sum_k L_k B_k - f_u \vec{r}_x \sum_k L_k B_k = f_u \vec{r}_x S_R + f_u t_x - t_z (u - u_0) - \vec{r}_z S_R (u - u_0) \quad (\text{C.23})$$

$$\sum_k L_k B_k ((u - u_0) \vec{r}_z - f_u \vec{r}_x) = S_R (f_u \vec{r}_x - \vec{r}_z (u - u_0)) + f_u t_x - t_z (u - u_0) \quad (\text{C.24})$$

and the same with the  $v$  component:

$$\sum_k L_k B_k ((v - v_0) \vec{r}_z - f_v \vec{r}_y) = S_R (f_v \vec{r}_y - \vec{r}_z (v - v_0)) + f_v t_y - t_z (v - v_0) \quad (\text{C.25})$$

Therefore, the system would only change its  $C$  matrix, including the  $S_R$  term, keeping the matrix  $A$  as it was:

$$C = \begin{pmatrix} f_u t_x - t_z \Delta u_1 + S_{R,1} (f_u \vec{r}_x - \vec{r}_z \Delta u_1) \\ f_v t_y - t_z \Delta v_1 + S_{R,1} (f_v \vec{r}_y - \vec{r}_z \Delta v_1) \\ \vdots \\ f_u t_x - t_z \Delta u_p + S_{R,p} (f_u \vec{r}_x - \vec{r}_z \Delta u_p) \\ f_v t_y - t_z \Delta v_p + S_{R,p} (f_v \vec{r}_y - \vec{r}_z \Delta v_p) \end{pmatrix} \quad (\text{C.26})$$